Hofmann, Stefan

Nr. 21 A

99326 Ilmtal OT Großhettstedt

0179/4628492

stefan.hofmann78@web.de

27163

Analyse von Metacomputing-Systemen zum verteilten Rechnen

Diplomarbeit

zur Erlangung des akademischen Grades
„Diplom-Wirtschaftsinformatiker“
an der Fakultät Informatik und Automatisierung
der Technischen Universität Ilmenau

Fachgebiet Telematik
Verantwortlicher Hochschullehrer: Prof. Dr. Ing. habil Dieter Reschke
Betreuer: Dipl.-Inf. Thorsten Strufe
Abgabetermin: 01.11.2003

Inhaltsverzeichnis

Abkürzungsverzeichnis

ACM	Association for Computing Machinery
BOINC	Berkeley Open Infrastructure for Network Computing
CBR	Cluster Based Routing Protocol
CC-NUMA	Cache-Coherent Nonuniform Memory Access
CPU	Central Processing Unit
DLT	Digital Linear Tape
DNS	Domain Name System
DSR	Dynamic Source Routing
E/A	Eingabe-Ausgabe
FTP	File Transfer Protocol
GByte	GigaByte
GHz	GigaHertz
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JVM	Java Virtual Machine
KHz	KiloHertz
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LOA	Legion Object Address
LOID	Legion Object Identifier
LRTL	Legion Runtime Library
MBit	MegaBit
MHz	MegaHertz
MIMD	Multiple Instruction Multiple Data
MISD	Multiple Instruction Single Data
MPL	Mentat Programming Language
NCC-NUMA	Non-Cache-Coherent Nonuniform Memory Access
NCSA	National Center für Supercomputing Anwendungen
NORMA	No Remote Memory Access
OGR	Optimal Golomb Ruler
OPA	Object Persistent Address

OPR	Object Persistent Representation
PC	Personal Computer
PDA	Personal Digital Assistant
P2P	Peer-to-Peer
SETI	Search for Extra Terrestrial Intelligence
SIMD	Single Instruction Multiple Data
SISD	Single Instruction Single Data
SMP	Symmetrische Multiprozessoren
TCP/IP	Transmission Control Protocol / Internet Protocol
VBS	Verteilte Betriebssysteme
VO	Virtuelle Organisation

Abbildungsverzeichnis

1 Einleitung

In diesem Kapitel werden Problemstellung, Zielsetzung und Aufbau der Arbeit dargelegt.

1.1 Problemstellung

Wohl kaum eine Technologie hat eine so rasante Leistungssteigerung erlebt wie die Computertechnologie.[1] In einer seit wenigen Jahrzehnten andauernden Entwicklung werden Prozessoren kontinuierlich immer schneller, wobei in den letzten Jahren, die „paralleles Rechnen ermöglichende Steigerung der Kommunikationskapazitäten von Verbindungsnetzwerken“[2] hinzukommt. So werden Computer in der Wissenschaft immer häufiger eingesetzt um den Menschen bei Forschungen zu unterstützen.[3] Die Anforderungen der Software an die Rechnerhardware sind trotz deren konstanter Weiterentwicklung in den letzten Jahren gestiegen und so treten zunehmend Anwendungsfälle auf, welche die Leistung einzelner Rechner überfordern. Dies sind beispielsweise Anwendungen zur Klimaforschung, zu Gen-Analysen, zu astrophysischen Simulationen, zu Störungssimulationen sowie zu Berechnungen von Börsenprognosen und Optimalsteuerproblemen. Die „übliche“ Lösung für dieses Problem sind so genannte Supercomputer. Diese sind allerdings sehr teuer, weshalb ihre Anschaffung vorrangig für finanzkräftige Institutionen lohnenswert ist.[4]

Aus diesem Ansatz entstand die Idee, dass es möglicherweise viel geschickter sein würde, das Problem in kleine Teile zu zerlegen und diese auf mehreren PCs zu berechnen, deren Leistungsfähigkeit folglich nicht so hoch sein muss.[5] Auch wenn sich durch die Aufteilung eines Problems normale Computer nutzen lassen, ist es den meisten Forschungseinrichtungen trotzdem nicht möglich, sich mehrere hundert oder sogar Tausende von PCs anzuschaffen. Vielmehr haben Computer häufig Rechenkapazität übrig, da der Rechner nur einen verschwindend geringen Teil seiner maximalen Leistungsfähigkeit ausnutzt, wenn der Anwender lediglich Briefe schreibt, der Bildschirmschoner „läuft“ oder sich der Nutzer in der Mittagspause befindet.

---

[1] o.V. /Rechnernetze/ 1

[2] o.V. /Rechnernetze/ 1

[3] Vgl. o.V. /Distributed Computing?/ 1

[4] Vgl. o.V. /Distributed Computing?/ 1

[5] Die Ausführungen des folgenden Abschnitts sind Vgl. o.V. /Distributed Computing?/ 1 entnommen worden.

Wäre es deshalb nicht eine gute Gelegenheit diese ungenutzte Kapazität zur Lösung wissenschaftlicher Probleme zur Verfügung zu stellen? Die durchzuführenden Berechnungen mit ausreichender Rechenleistung zu versorgen ist der Ausgangspunkt der Kopplung mehrerer Rechner zu einem Metacomputer. Der Begriff des „Metacomputing“ wurde erstmalig im Jahr 1992 in einem Artikel der „Communication of ACM“ der Autoren *Charles Catlett* und *Larry Smarr* vom National Center für Supercomputing der Universität von Illinois eingeführt.[6] Sie definieren diesen Begriff als „a network of heterogeneous, computational resources linked by software in such a way that they can be used as easily as a personal computer“[7]. Das ursprüngliche Ziel war es, einen Computerverbund aufzubauen, bei dem, vergleichbar mit einem Stromnetz, jeder Nutzer an jedem Ort seinen Computer über ein Kabel mit einer Steckdose verbindet, um somit weltweit auf die Computerressourcen zugreifen zu können, die zur Bewältigung seiner Problemstellung benötigt werden.[8] „Das Computernetz wird immer mehr zum Computer“[9]

Mit der stetigen Verbreitung des Internets sind seit längerer Zeit auch die technischen Grundlagen vorhanden, um solche Projekte durchzuführen. Dafür stellt das Forschungsteam einen Server im Internet bereit, von dem sich die potentiellen Nutzer ein Programm, einen so genannten Client, auf den eigenen PC herunterladen können. Sobald dieser Client aktiviert wurde, verbindet er sich mit dem Server im Internet und lädt von dort ein Datenpaket[10], welches die auszuführenden Arbeitsanweisungen enthält. Bei diesem handelt es sich lediglich um ein geringfügiges Teilproblem der Gesamtaufgabe, an welchem der Client bis zur Fertigstellung arbeitet und die Ergebnisse der Berechnung anschließend zurück an den Server des Forschungsteams schickt, um ein weiteres Datenpaket anzufordern.

Diese und andere Probleme bescheren der Computerwelt eine Menge an Wortschöpfungen, wobei viele Begriffe firmen- und produktspezifischer Prägung sind und vorrangig dem Produktmarketing dienen.

---

[6] Vgl. Schirmbacher /Metacomputing-Editorial/ 1

[7] Schirmbacher /Metacomputing-Editorial/ 1

[8] Vgl. Schirmbacher /Metacomputing-Editorial/ 1

[9] Schirmbacher /Metacomputing-Editorial/ 1

[10] Diese Datenpakete werden auch als Work Unit bezeichnet. Vgl. o.V. /Distributed Computing?/ 1

1.2 Ziel der Arbeit

Das Ziel dieser Arbeit ist die Auseinandersetzung mit der Technologie des „Metacomputing“, welches in den letzten Monaten und Jahren zunehmend in den Mittelpunkt der Forschung gerückt ist. Dabei wird ausgehend von grundlegenden Begriffen, Definitionen und Technologien im Bereich des verteilten Rechnens auf den Begriff des Metacomputing hingeführt. Anschließend wird eine Beschreibung und Analyse gegenwärtiger Initiativen und Projekte des Metacomputing vorgenommen und die Problematik der Sicherheit von Metacomputing-Systemen beleuchtet.

1.3 Aufbau der Arbeit

Das zweite Kapitel befasst sich einführend mit der Thematik Ad-hoc-Netzwerke als eine Möglichkeit der drahtlosen Kommunikation. In Kapitel drei werden die Merkmale verteilter Systeme näher beleuchtet und anschließend verschiedene Topologien dieser vorgestellt. Im vierten Kapitel wird eine Abgrenzung des Parallelen vom Verteilten Rechnen in Form einer Gegenüberstellung beider Konzepte und deren Architekturen vorgenommen. In Kapitel fünf werden Zentrale und Dezentrale Kommunikationsmodelle, anhand verteilter Betriebssysteme für Zentrale und des Peer-to-Peer-Konzeptes für Dezentrale Kommunikationsmodelle, unterschieden. Im sechsten Kapitel wird die Grid-Technologie als eine neue Netzstruktur zur Bewältigung komplexer Probleme erläutert, um Ressourcen zu verteilen oder zu bündeln und brachliegende Ressourcen für sinnvolle Aufgaben zu nutzen. In Kapitel sieben werden die Technologie des Metacomputing sowohl im Allgemeinen als auch im Speziellen beleuchtet und die Anforderungen an Metacomputing-Systeme dargelegt. Darüber hinaus werden die Dienste von Metacomputing-Systemen beschrieben und das Konzept des Metacomputing in den Kontext des Parallelen und Verteilten Rechnens eingeordnet. Im achten Kapitel wird auf ausgewählte Systeme aus dem Spektrum gegenwärtig existierender Metacomputing-Systeme näher eingegangen. In Kapitel neun stehen der Sicherheitsgedanke an sich im Bereich des wissenschaftlich-technischen Rechnens sowie dessen Bedeutung und Konsequenzen für die in Kapitel acht vorgestellten Metacomputing-Systeme im Mittelpunkt.

2 Ad-hoc-Netzwerke

In diesem Kapitel möchte ich „Ad-hoc-Netzwerke“ vorstellen. Nach einer Einführung in die Thematik werde ich, ausgehend von einem Überblick über Routingalgorithmen

selbst organisierender Netzwerke, das „Cluster Based Routing“-Protokoll beleuchten und in diesem Zusammenhang auf das Problem der Sicherheit eingehen.

2.1 Einführung in Mobile Ad-hoc-Netzwerke

Mit der kontinuierlich wachsenden Nutzung des Internet steigen auch die Forderungen drahtloser Zugangsmöglichkeiten an. So wächst die Anzahl der Benutzer mit mobilen Endgeräten wie z.B. Notebook, PDA, Handy derzeit weit schneller als die an das Internet angeschlossenen Rechner.[11] „Mobile Ad-hoc-Netzwerke“, kurz „MANET“[12] sind in den letzten Jahren zunehmend zentraler Bestandteil der Forschungsaktivitäten geworden und können zukünftig eine sehr bedeutende Rolle im Bereich verteilter Systeme spielen.[13] „Mobile Ad-hoc-Netzwerke bilden sich spontan ohne jegliche vorhandene Infrastruktur und ohne Administration aus autonomen mobilen Systemen.“[14] Es sind drahtlose, sich schnell entwickelnde Netzwerke, die die Fähigkeit besitzen, sich jederzeit selbst zu organisieren ohne dabei eine feste Infrastruktur wie beispielsweise Router oder Basisstationen zu benötigen.[15] Sie sind in der Lage, Pakete an unterschiedliche mobile Endgeräte weiterzuleiten.[16] Die mobilen Teilnehmer organisieren sich in dem Netzwerk selbst und können sich relativ frei und mehr oder weniger schnell bewegen, ohne den Kontakt zum Netz zu verlieren; denn die Knoten selbst bilden gemeinsam das Netz. Dabei können die benutzten Geräte ein- und ausgeschaltet werden, wodurch sich die Verbindung der Geräte untereinander ständig ändert.[17]

Im Gegensatz zu fest verdrahteten Netzwerken ist ein mobiles Ad-hoc-Netzwerk äußerst dynamisch aber mit sehr geringer Bandbreite, wodurch sich andere Protokollanforderungen für die Kommunikation ergeben.[18]

Die besondere Herausforderung in einem mobilen Ad-hoc-Netzwerk ist die Wegefindung[19] vom Sender zum Empfänger aufgrund der Dynamik der Topologie. Durch die

---

[11] Vgl. o.V. /Projektgruppe/

[12] Vgl. Kaldewey /Simulation/, Vgl. o.V. /Projektgruppe/

[13] Vgl. McDonald, Znati /A Path Availability Model/

[14] Kaldewey /Simulation/

[15] Vgl. McDonald, Znati /A Path Availability Model/ , Vgl. Günther /Einführung in Ad-hoc Netzwerke/

[16] Vgl. Jansen / Routingalgorithmen/ 10, Vgl. Günther /Einführung in Ad-hoc Netzwerke/

[17] Vgl. o.V. /Ad-hoc-Netzwerke/

[18] Vgl. o.V. /Projektgruppe/

[19] Diese Wegefindung wird in der Literatur auch als Routing bezeichnet. Vgl. o.A. /Projektgruppe/

Mobilität der Knoten kann sich die Topologie des Systems jederzeit ändern, woraus erheblich komplexere Anforderungen an den Routingalgorithmus entstehen.[20]

2.2 Funktionsweise eines Mobilen Ad-hoc-Netzwerkes

„Jeder Knoten hat Peer-to-Peer-Verbindungen zu allen Knoten in seiner unmittelbaren Umgebung.“[21] Aus der Gesamtheit dieser Knoten bildet sich spontan und sehr dynamisch ein Netz, das kontinuierlichen Veränderungen unterliegt.[22] So ist es beispielsweise möglich, dass ein sich ständig in Bewegung befindender Knoten den Kontakt zu seinem derzeitigen Nachbarknoten verliert aber an einer anderen Stelle wieder Anschluss zum Netz findet. Damit die für den entsprechenden Knoten vorgesehenen Pakete diesen auch nach der Änderung seiner Position erreichen können, gibt es sich selbst organisierende Routingprotokolle und -algorithmen, die die Fähigkeit haben, sich an die veränderlichen Gegebenheiten anzupassen. Die Datenübertragung an sich funktioniert über so genannte „Multihop-Wireless-Links“, bei denen ein Datenpaket von Knoten zu Knoten solange weitergereicht wird, bis es den Empfänger erreicht. Dabei müssen die mobilen Endgeräte auch als Router agieren, was in fest verdrahteten Netzen nicht der Fall ist. Ein Problem in Ad-hoc-Netzwerken ist die Abhängigkeit der mobilen Geräte von Batterien. Um die Laufzeit dieser zu erhöhen, wird gewöhnlich die Funkleistung gesenkt, was jedoch das Gesamtnetz schwächt oder sogar zerfallen lässt. Darüber hinaus ist die benötigte Bandbreite oft geringer, als für die Bedürfnisse der Nutzer notwendig wäre.

2.3 Ad-hoc Routingalgorithmen

In der Fachliteratur werden Algorithmen für den Einsatz in selbst konfigurierenden Netzwerken, den Ad-hoc-Netzwerken, anhand ihres Funktionsprinzips in Algorithmen für flaches Routing und in hierarchische Algorithmen unterschieden.[23]

„Im flachen Routing sind die Funktionen homogen verteilt, die einzelnen Stationen unterscheiden sich in ihrem Funktionsumfang.“[24] Durch die beliebige Austauschbarkeit der Stationen wird das Routing erheblich vereinfacht. Um die hohe Anzahl flacher Routingalgorithmen klassifizieren zu können, werden zusätzlich reaktive und proaktive Al-

---

[20] Vgl. o.A. /Projektgruppe/

[21] Hohenberger /Sicherheit/

[22] Die folgenden Ausführungen sind Vgl. Hohenberger /Sicherheit/ entnommen worden.

[23] Vgl. Jansen / Routingalgorithmen/ 36

[24] Jansen /Routingalgorithmen/ 38

gorithmen sowie die für Sonderfälle geschaffene Klasse des botschaftenlosen Routing unterschieden. Während die reaktiven Algorithmen erst dann eine Route suchen, wenn ein Bedarf dafür besteht, versuchen proaktive Algorithmen kontinuierlich ihre Routen an das aktuelle Netzwerk anzupassen, weshalb immer aktuelle Routen zur Verfügung stehen.[25]

„Das Kriterium für hierarchisches Routing ist die inhomogene Funktionsaufteilung unter den Stationen, dabei übernehmen einzelne Stationen spezialisierte Funktionen innerhalb einer Gruppe. Eine Aufteilung der Funktionen ist besonders vorteilhaft, wenn große Netzwerke verwaltet werden müssen, da der Verwaltungsaufwand in großen Netzwerken überproportional anwächst.“[26]

Im Folgenden werde ich auf das „Cluster Based Routing Protocol“, kurz CBR, als einen hierarchischen Routingalgorithmus näher eingehen.[27]

Das CBR-Protokoll teilt die Stationen eines Netzwerkes in sich teilweise überlappende Gruppen auf, die als Cluster bezeichnet werden. So ist es möglich, dass Stationen im Randbereich eines Clusters mehreren Gruppen angehören. Diese Gruppen bilden sich aus einer Ansammlung von Stationen, die sich zuerst einen so genannten „Clusterhead“ wählen. Ab diesem Zeitpunkt sind alle direkten Nachbarn der Station diesem Cluster zugeordnet. Die Beziehungen der Stationen untereinander müssen durch kontinuierlich ausgesendete Kennungen überprüft werden, um die Cluster gegebenenfalls neu zu organisieren. Eine Route wird ähnlich wie beim „Dynamic Source Routing“[28], kurz DSR, durch Fluten[29] gesucht, wobei die notwendigen Botschaften lediglich über die Clusterheads geschickt werden. Dieses Vorgehen reduziert insbesondere in dichten Netzwerken die Anzahl der Botschaften für das Fluten. Das CBR-Protokoll kann innerhalb des Clusters als proaktiv und außerhalb als reaktives Protokoll bezeichnet werden. Der Vorteil der Gruppenbildung ist, dass das Fluten sehr schnell und effizient ausge-

---

[25] Vgl. Jansen /Routingalgorithmen/ 38

[26] Jansen /Routingalgorithmen/ 38

[27] Die folgenden Ausführungen sind Vgl. Bechler et.al. /Sicherheitskonzept/, Vgl. Jansen /Routingalgorithmen/ 46 entnommen worden.

[28] Das Dynamic Source Routing war der erste speziell für Ad-hoc-Netzwerke entwickelte Routingalgorithmus. DSR ist laut der Definition in der Fachliteratur ein flacher, reaktiver Algorithmus, da gezielt nach benötigten Pfaden gesucht wird. Vgl. Jansen /Routingalgorithmen/ 39

[29] Das Fluten von Daten stellt die einfachste Art des Routing dar. Dabei werden die Pakete an alle Stationen im Netzwerk verteilt. Das Fluten ist eine Methode, die ohne Wissen über die Topologie des Netzwerkes angewandt werden kann. Darüber hinaus verwendet es stets den kürzesten Pfad, da es jeden benutzen kann. Vgl. Jansen /Routingalgorithmen/ 33

führt werden kann; einen wesentlichen Nachteil stellt der Verwaltungsaufwand für die Bildung und Pflege dieser Gruppen dar.

2.4 Sicherheit in Ad-hoc-Netzwerken

Ad-hoc-Netzwerke sind Angriffen hinsichtlich der Sicherheit gegenwärtig nahezu schutzlos ausgeliefert. Die drahtlose Kommunikation kann einerseits durch passive Angriffe abgehört werden und ist andererseits durch aktive Angriffe auf die Kommunikationsprotokolle verwundbar.[30] Durch die spezifischen Eigenschaften von Ad-hoc-Netzen wie Dynamik, begrenzte Bandbreite, störanfällige Verbindungen u.a. wird der Einsatz entsprechender Sicherheitsmechanismen zusätzlich erschwert. Nachfolgend werde ich näher auf die Sicherheitsanforderungen an mobile Ad-hoc-Netzwerke eingehen, an die nahezu die gleichen Anforderungen wie an allgemeine Sicherheitskonzepte gestellt werden, wobei der Aspekt der Verfügbarkeit speziell für Ad-hoc-Netzwerke eine wichtige Rolle spielt.[31]

Die „Authentizität“ bezeichnet den sicheren Nachweis der Identität des beteiligten Kommunikationspartners beziehungsweise der Herkunft der übertragenen Daten und ist eine wesentliche Grundlage für die Sicherheit in Kommunikationsnetzen.

Darüber hinaus muss ein sicheres System die „Integrität“ der übertragenen Daten enthalten, denn die Nachricht soll unverändert das Ziel erreichen. So sollen zum einen mutwillige Manipulationen und zum anderen zufällige Übertragungsfehler verhindert oder zumindest erkannt werden.

Die Forderung nach „Vertraulichkeit“ beschreibt den Schutz der Daten vor nichtautorisierten Personen oder Geräten. Diese stellt eine große Herausforderung in Ad-hoc-Netzen dar, da die vorwiegend verwendete Luftschnittstelle nicht abgesichert werden kann und sich somit potentielle Angreifer im Funkradius eines an der Kommunikation beteiligten Knotens befinden können.

Das Ziel der „Nichtabstreitbarkeit“ bezeichnet die eindeutige Nachweisbarkeit gegenüber an der Kommunikation beteiligten Dritten. Dabei wird zwischen der Nichtabstreitbarkeit des Sendens, des Empfangens und der Übermittlung unterschieden.

Unter „Autorisierung“ verbirgt sich die Vergabe von Rechten zum Zugriff auf Ressourcen oder zum Ausführen entsprechender Aktionen.

---

[30] Vgl. Bechler et.al. /Sicherheitskonzept/

[31] Die nachfolgenden Ausführungen sind Vgl. Hohenberger /Sicherheit/ und Bechler et.al. /Sicherheitskonzept/ entnommen worden.

Der in herkömmlichen Netzen nahezu unwichtige Aspekt der „Verfügbarkeit“ ist in mobilen Netzwerken umso wichtiger, um die ständige Präsenz von Diensten und Ressourcen zu gewährleisten. Da in Ad-hoc-Netzen jeder Knoten gleichzeitig auch ein Router ist, kann der Ausfall eines Knotens erhebliche Auswirkungen auf das Netz haben.

2.5 Schlussfolgerungen für das Metacomputing

Dieses Konzept der Ad-hoc-Netzwerke könnte in naher Zukunft auch eine Vision für das Verteilte Rechnen im Zusammenhang mit Metacomputing (Vgl. Kapitel 7) darstellen. So könnten die an einem Ad-hoc-Netzwerk beteiligten Rechner neben den herkömmlichen Aufgaben im Netzwerk, auch Herausforderungen für das wissenschaftlich-technische Rechnen übernehmen.

Einen wesentlichen Vorteil würde diese Möglichkeit der Vernetzung mobiler Endgeräte für Teilnehmer mit so genannten „schwachen“ Knoten wie Handy, PDA u.a. bieten, um komplex zu bewältigende Aufgaben, wie z.B. Ver- und Entschlüsselung von Daten, auf leistungsfähigere und performancestärkere Knoten wie beispielsweise Notebooks auszulagern. Das würde es Nutzern mit vergleichsweise leistungsschwächeren Endgeräten ermöglichen, in einem angemessenen Zeitrahmen und unter verbesserten Bedingungen zu arbeiten. Dies könnte so funktionieren, indem die zu bearbeitenden Aufgaben den in einem entsprechenden Ad-hoc-Netz drahtlos verknüpften Knoten mitgeteilt werden. Sobald die potentiellen Teilnehmer ihre Bereitschaft zur Mitarbeit erklärt haben, wird zunächst ein Client herunter geladen, dann die für die Berechnung benötigten Datenpakete transferiert und im Anschluss die eigentliche Berechnung gestartet. Unter Verwendung verteilter Betriebssysteme hingegen ist es nicht notwendig ganze Clients herunter zu laden, da diese Systeme die Möglichkeit bieten, einzelne Tasks selbst zu verteilen. (Vgl. Kapitel 5.1) Während der Berechnung müssen die Teilnehmer nicht zwingend in dem ursprünglichen Ad-hoc-Netzwerk verfügbar sein. Ist jedoch die Berechnung abgeschlossen und die Teilnehmer sind wieder im anfänglichen Netz verfügbar, können die bereits berechneten Datenpakte zurücktransferiert werden, um anschließend neue Datenpakete anzufordern.

3 Verteilte Systeme

Im folgenden Kapitel wird das Konzept verteilter Systeme näher beleuchtet. Dabei werden sowohl deren wesentliche Merkmale als auch mögliche Topologien dargestellt.

3.1 Definition und Merkmale von verteilten Systemen

Nach *Tanenbaum* werden die Begriffe „verteiltes System“ und „Rechnernetz“ in der Literatur häufig miteinander verwechselt, wobei der eigentliche Unterschied darin besteht, dass die Existenz mehrerer autonomer[32] Rechner in einem verteilten System für den Benutzer transparent ist.[33] Er verwendet den Begriff Rechnernetz „für mehrere miteinander verbundene autonome Computer“[34]. Ein verteiltes System ist nach *Tanenbaum* „eine Menge voneinander unabhängiger Computersysteme, die dem Benutzer den Eindruck vermitteln, es handle sich um einen einzigen Computer“[35]; er versteht dieses als „ein besonderes Netz, bei dem die Software dem System ein hohes Maß an innerer Geschlossenheit und Transparenz verleiht“[36]. Folglich besteht der Unterschied zwischen einem verteilten System und einem Rechnernetz vorrangig in der Software, speziell im Betriebssystem, und weniger in der Hardware.[37]

Coulouris definiert ein verteiltes System mit: “A distributed system consists of a collection of autonomous computers linked by a computer network and equipped with distributed system software.“[38] Demnach bilden die Rechner eines Netzwerkes ein verteiltes System, wenn ihnen eine geeignete Software zur Verfügung steht.[39] Diese Software soll es den Rechnern ermöglichen, ihre Aktivitäten miteinander zu koordinieren und die Ressourcen des gesamten Systems gemeinsam zu nutzen.[40] Als Ressourcen werden dabei Hardware, Software und Daten verstanden. Ein relativ einfaches Beispiel eines verteilten Systems wäre beispielsweise ein Raum, in dem sich mehrere Workstations befinden, von denen eine als Server fungiert.[41] Diese Workstation verwaltet eine Festplatte sowie Dateien und Software, die allen anderen Rechnern zur Verfügung stehen. Außer-

---

[32] Als autonome Rechner werden Computer bezeichnet, die wiederum andere Computer beliebig ein- oder ausschalten bzw. steuern können und somit als voneinander unabhängig bezeichnet werden. Vgl. Tanenbaum /Computernetzwerke/ 18

[33] Vgl. Tanenbaum /Computernetzwerke/ 18

[34] Tanenbaum /Computernetzwerke/ 18

[35] Karl /Kommunikation/

[36] Tanenbaum /Computernetzwerke/ 18

[37] Vgl. Tanenbaum /Computernetzwerke/ 18

[38] Coulouris, Dollimore, Kindberg /Concepts and Design/ 2

[39] Vgl. Schneider, Quandt /Grundlagen/

[40] Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 2

[41] Die folgenden Ausführungen sind Vgl. Schneider, Quandt /Grundlagen/ entnommen worden.

halb dieses Raumes befinden sich weitere Server wie z.B. Mailserver oder Drucker, die ebenfalls allen Rechnern und deren Nutzern zur Verfügung stehen.
Ein verteiltes System hat verschiedene Merkmale. Sechs davon werden in der Fachliteratur als wesentlich angesehen, damit ein verteiltes System sinnvoll genutzt werden kann.[42] Nachfolgend möchte ich näher auf diese Merkmale eingehen.

3.1.1 Ressourcen-Teilung (resource-sharing)

Das Prinzip der Ressourcen-Teilung bezeichnet die Freiheit des Zugriffs auf alle nicht privaten Ressourcen des verteilten Systems von allen Rechnern des Systems aus.[43] Ressourcen sind dabei Gegenstände, die zur Nutzung verfügbar sind und durch die drei Bereiche Hardware (Drucker, Festplatten, Prozessoren), Software (Compiler, Editoren, Bibliotheken) und Daten (Dateien, Datenbanken) repräsentiert werden. Die Nutzer dieser Ressourcen können Softwareentwickler, Nutzer von Anwendungen u.a. sein. Der Nutzer einer Ressource kann als Prozess betrachtet werden, der auf diese zugreifen will. Dieser Zugriff wird durch den Ressourcen-Verwalter (resource manager), ein Softwaremodul zur Verwaltung von Ressourcen eines gegebenen Typs, gesteuert. Die Gestaltung des Ressourcenzugriffs erfolgt durch die Prozesskommunikation zwischen dem Ressourcen-Verwalter und dem Nutzer mittels unterschiedlicher Modelle. Dies kann durch das einfache, robuste und weit verbreitete „Client-Server Modell“, durch das elegante und konzeptionell einfache „Objekt-basierte Modell“ aber auch mittels „mobiler Agenten“ oder über gleichrangige Prozesse wie im „Peer-2-Peer“ geschehen.

3.1.2 Offenheit (openness)

Ein verteiltes System wird als offen bezeichnet, wenn es durch neue Ressourcen wie beispielsweise Hard- und Software ohne Nebenwirkungen auf die existierenden Komponenten erweitert werden kann.[44] Im Allgemeinen basiert diese Offenheit von verteilten Systemen „auf der Publizierung ihrer Schnittstellen und der Gestaltung eines einheitlichen Mechanismus zur Prozesskommunikation, durch den auf diese Schnittstellen zugegriffen werden kann“[45]. Demzufolge können verteilte Systeme aus Komponenten

---

[42] Vgl. Schneider, Quandt /Grundlagen/

[43] Die folgenden Ausführungen wurden Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 10,11 und Vgl. Schneider, Quandt /Grundlagen/ entnommen.

[44] Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 14

[45] Schneider, Quandt /Grundlagen/

verschiedener Hersteller zusammengesetzt werden, was einen beachtlichen Zuwachs an modernsten Technologien ermöglichen kann. Ein Beispiel eines offenen Systems ist das Betriebssystem „Unix“. Auf dessen Ressourcen kann über so genannte „System Calls“ zugegriffen werden, die undokumentiert sind und den Anwendungsentwicklern zur Verfügung stehen.[46] Darüber hinaus kann Unix aufgrund seiner Hardwareunabhängigkeit erweitert werden, so dass es offen für Hardware-Hersteller und Systemverwaltungssoftware ist.

3.1.3 Gleichzeitigkeit (concurrency)

Die Möglichkeit zur gleichzeitigen Bedienung von mehreren Nutzern, Anwendungen oder Prozessen in einem verteilten System ergibt sich aus der Tatsache, dass das Merkmal der Ressourcen-Teilung (Vgl. Kapitel 3.1.1) als Grundeigenschaft vorliegen muss.[47] Auf dieser Basis entsteht Gleichzeitigkeit zum einen durch den Zugriff mehrerer Nutzer auf dieselbe Ressource und zum anderen stellen mehrere Dienstanbieter unterschiedlichen Nutzern dieselbe Ressource zur Verfügung.[48] Der Zugriff auf dieselbe Ressource durch mehrere Nutzer ist auch außerhalb der Thematik verteilter Systeme allgemein bekannt, denn der Prozessor eines Rechners kann ebenfalls von allen Nutzern des Systems genutzt werden.[49] Bei vom Dienstanbieter für unterschiedliche Nutzer zur Verfügung gestellten Ressourcen tritt Gleichzeitigkeit dann auf, wenn beispielsweise eine Ressource von mehreren Servern bedient wird, die jeweils die Aufgaben unterschiedlicher Nutzer erledigen, um sich gegenseitig zu entlasten. Dabei müssen die Server derselben Ressource so gestaltet werden, dass genau ein Server den Job eines Nutzers annimmt und Zustandsänderungen der Ressource allen Servern bekannt gemacht werden.

3.1.4 Skalierbarkeit (scalability)

Ein wesentlicher Vorteil verteilter Systeme ist die Erweiterbarkeit dieser bei Bedarf, indem zusätzliche Rechnerkapazitäten hinzugefügt werden können. Demnach soll bei einer solchen Skalierung eine Änderung der Systemsoftware als auch der Anwendungen verhindert werden.[50] Dabei muss berücksichtigt werden, dass die Skalierung nicht nur

---

[46] Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 15

[47] Vgl. Schneider, Quandt /Grundlagen/

[48] Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 16

[49] Die folgenden Ausführungen wurden Vgl. Schneider, Quandt /Grundlagen/ entnommen.

[50] Die folgenden Ausführungen sind Vgl. Schneider, Quandt /Grundlagen/ entnommen worden.

ein Problem der Leistung an sich ist, denn bei der Skalierung eines Systems erhöht sich vor allem die Belastung der Systemressourcen. Die bislang erfolgreichste Lösung solcher Probleme ist die Replizierung. So werden ausgelastete Ressourcen und Dienste repliziert und unter der Philosophie entworfen, dass alle Ressourcen unendlich belastbar sind, da das System unendlich erweitert werden kann. Die Verwirklichung dieser Lösung stößt jedoch immer öfter an Grenzen, wie z.B. durch die zunehmende Belastung der Netzkapazität im Internet, Hardwarebeschränkungen, der exponentiell steigenden Teilnehmer im Internet sowie der gleichzeitig sinkenden Anzahl verfügbarer IP-Adressen.

3.1.5 Fehlertoleranz (fault tolerance)

Fehler werden als Ereignisse bezeichnet, die den ordnungsgemäßen Ablauf des Systems beeinträchtigen. Treten Fehler in Rechnersystemen auf, können Datenverluste sowie fehlerhafte Abläufe oder Unterbrechungen von Anwendungsprogrammen entstehen.[51] Die Eigenschaft eines Systems, beim Eintreten solcher Fehler seine Dienste weiter anbieten zu können, wird als Fehlertoleranz bezeichnet.[52] Diese kann teuer sein, da beispielsweise die Anschaffung redundanter Hardware finanziell kostspielig und vor allem dann überflüssig ist, wenn die normale Belastung des Systems zu gering ist, um sie auszulasten. Die Fehlertoleranz hinsichtlich der Software bedeutet die Nutzung von Informationen, die für den Systemablauf redundant sind und für die demzufolge ein erhöhter Speicherplatzbedarf entsteht. Deshalb muss Fehlertoleranz gegen die Systemleistung abgewogen werden, um ein realistisches Verhältnis zwischen hoher Leistung und der Toleranz von Fehlern zu erreichen.

3.1.6 Transparenz (transparency)

„Ein verteiltes System ist transparent, wenn die Verteilung, also die Tatsache, dass das verteilte System aus Komponenten besteht, vom Nutzer bzw. Anwendungsprogramm nicht wahrgenommen wird. Der Nutzer nimmt dagegen das verteilte System als eine Einheit wahr.“[53] Transparenz kann auf unterschiedlichen Ebenen auftreten, so dass verschiedene Arten von Transparenz entstanden sind.[54] So gibt es die so genannte „Access

---

[51] Vgl. Coulouris, Dollimore, Kindberg /Concepts and Design/ 19

[52] Die folgenden Ausführungen sind Vgl. Schneider, Quandt /Grundlagen/ entnommen worden.

[53] Schneider, Quandt /Grundlagen/

[54] Die folgenden Ausführungen wurden Vgl. Schneider, Quandt /Grundlagen/ entnommen.

transperency“ für den Zugriff auf lokale und ferne Objekte durch identische Operationen, die „Location transperency“, welche die Angabe des Standortes beim Zugriff auf das Objekt nicht erfordert oder die „Concurrency transperency“, bei der mehrere Prozesse auf dasselbe Objekt zugreifen können, ohne die Richtigkeit des Ablaufs zu beeinträchtigen. Weitere Arten der Transparenz sind “Failure-“, „Migration-“, „Performance-“ oder „Scaling-transperency“.

3.2 Topologien verteilter Systeme

Während das Internet selbst das größte dezentrale Computersystem der Welt ist, ist die überwiegende Anzahl der seit 1990 entwickelten Systeme im Internet zentralisiert.[55] Die kontinuierlich fortschreitende Entwicklung von Peer-to-Peer-Netzwerken (Vgl. Kapitel 5) auf Basis der so genannten „Gnutella“[56]-Architektur und deren propagierte Robustheit, Offenheit und inhärente Lastverteilung führt gegenwärtig zu einem Aufschwung dezentraler Systeme. Der Unterschied von zentralen und dezentralen Architekturen beruht auf deren Topologie - der Art wie Computer oder Programme im jeweiligen System verknüpft sind. Im Nachfolgenden möchte ich auf die vier Basistopologien verteilter Systeme eingehen. Die sind: Zentrale-, Ring-, Hierarchische- und Dezentrale Architektur.

Die in Abb. 3-1 dargestellte zentrale Architektur ist die wohl bekannteste Form und wird typischerweise als Client-Server-Modell in Datenbanken, Webservern und anderen einfachen verteilten Systemen eingesetzt.

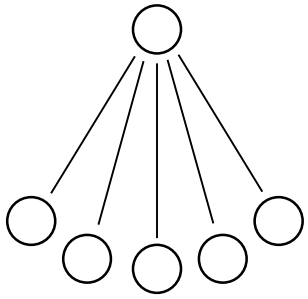

Abb. 3-1: Zentrale Architektur[57]

Dabei werden alle Funktionen und Informationen zentral auf einem Server verwaltet, der die Informationen an die mit ihm verbundenen Clients sendet bzw. von ihnen

[55] Die folgenden Ausführungen sind Vgl. Minar /Topologies/ entnommen worden.

[56] Gnutella ist ein Peer-to-Peer Protokoll und stellt somit die Basis des Gnutella-Netzwerkes.

[57] Vgl. Minar /Topologies/

empfängt. Viele der als „Peer-to-Peer“ bezeichneten Anwendungen sind mit einer zentralen Komponente ausgestattet.

Aufgrund der Tatsache, dass ein einzelner zentraler Server die Anfragen der Clients nur schwer bewältigen kann, bietet es sich an, mehrere Rechner als Cluster in einer Ringstruktur, wie in Abb. 3-2, zu verknüpfen, um so das Verhalten eines verteilten Servers entstehen zu lassen.

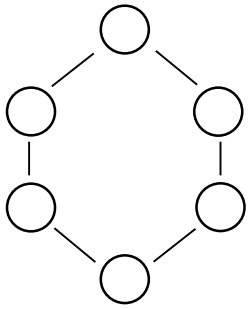

Abb. 3-2: Ringarchitektur[58]

Dabei führen einige der Rechner zum Teil ähnliche oder sogar gleiche Funktionen aus. Die daraus entstehenden Vorteile, größere Fehlertoleranz und bessere Lastverteilung, sind wesentliche Eigenschaften dieser Architektur.

Die Hierarchische Struktur in Abb. 3-3 hat sich über Jahre in der Geschichte des Internet entwickelt, wurde in der Praxis allerdings oft als eigene und klar ausgeprägte Topologie verteilter Systeme verkannt.[59]

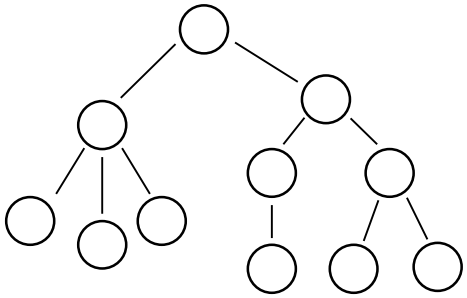

Abb. 3-3: Hierarchische Architektur[60]

Das wohl am besten bekannteste hierarchische System im Internet ist das Domain Name System, kurz DNS, zur Auflösung der Domain Namen zu den entsprechenden IP-

[58] Vgl. Minar /Topologies/

[59] Die folgenden Ausführungen sind Vgl. Freismuth /P2P-Zukunftsmodell/ entnommen worden.

[60] Vgl. Minar /Topologies/

Adressen. Dafür wird in jeder Hierarchieebene ein Name-Server für den entsprechenden Namen gesucht.

In Abb. 3-4 ist die dezentrale Architektur dargestellt, bei der alle Rechner symmetrisch kommunizieren und die gleiche Rolle im Netzwerk einnehmen.[61]

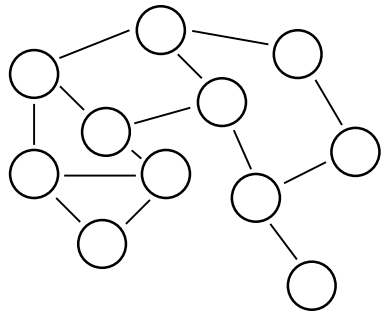

Abb. 3-4: Dezentrale Architektur[62]

Das Gnutella-Netzwerk ist derzeit vermutlich das am meisten dezentralisierte Netzwerk in der Praxis, mit lediglich einer zentralisierten Funktion, der Aufnahme eines neuen Hosts in das Netzwerk. Neben dem Gnutella-Netzwerk gibt es eine Reihe weiterer Filesharing-Systeme mit dezentraler Struktur wie beispielsweise „Freenet“ oder „OceanStore“, um nur einige zu nennen. Darüber hinaus ist die Architektur des Internet-Routing weitestgehend dezentralisiert durch das Border Gateway Protocol zur Koordination der Peers zwischen den verschiedenen Systemen.

In der Praxis haben verteilte Systeme oft eine komplexere Organisation als in den zuvor dargestellten Basistopologien beschrieben.[63] Folglich werden einzelne dieser Topologien miteinander kombiniert und als „Hybride Topologien“ bezeichnet. Die beteiligten Rechner innerhalb dieser Hybriden Systeme können verschiedene Rollen einnehmen und somit in einem Teil des Netzwerkes eine zentrale Funktion übernehmen, während sie gleichzeitig Teil einer Hierarchie in einem anderen Teil des Netzwerks sein können.

Bei der Kombination „Zentralisiert & Ring“ handelt es sich aus Sicht der Clients um ein zentrales System, welches die Robustheit einer Ringstruktur aufweist und vorrangig für leistungsstarke Webserveranwendungen genutzt wird. Da in einem zentralisierten System ein Server oftmals einen Client von einem oder mehreren anderen Servern darstellt,

---

[61] Vgl. Freismuth /P2P-Zukunftsmodell/

[62] Vgl. Minar /Topologies/

[63] Die folgenden Ausführungen sind Vgl. Freismuth /P2P-Zukunftsmodell/ und Vgl. Minar /Topologies/ entnommen worden.

wird die Verknüpfung „Zentralisiert & Zentralisiert“ genutzt, um verschiedene Funktionalitäten zu kombinieren.

Eine neue Architektur dezentraler Systeme ist die Einbettung zentralisierter Systeme in ein dezentrales System als Kombination „Zentralisiert & Dezentralisiert“. Diese hybride Topologie mit mehreren Tausend Peers ist beispielsweise im „FastTrack“-Filesharing-System realisiert, welches von „Kazaa“[64] und „Morpheus“[65] benutzt wird. Dabei hat die Mehrzahl der Peers eine zentralisierte Beziehung zu einem Superrechner, der alle Datenanfragen zu diesem Server weiterleitet. Anstatt diese Superrechner als einzelne Server arbeiten zu lassen, werden diese zu einem „Gnutella“-ähnlichen dezentralen Netzwerk verbunden, um die Anfragen zu verteilen.

Eine weitere Anwendung dieser hybriden Topologie ist die Verwendung von Internetemails. Die Mailclients haben eine zentralisierte Beziehung zu einem spezifischen Mailserver. Diese Server wiederum verteilen die E-Mails in dezentraler Art und Weise.

4 Paralleles Rechnen vs. Verteiltes Rechnen

In diesem Kapitel möchte ich eine Abgrenzung der Konzepte des Parallelen Rechnens gegenüber dem Verteilten Rechnen vornehmen. Dabei setze ich während dieser Arbeit Paralleles Rechnen mit Parallel Computing sowie Verteiltes Rechnen mit Distributed Computing gleich.

4.1 Definition und Gegenüberstellung beider Konzepte

In der Literatur werden die Begriffe Paralleles Rechnen und Verteiltes Rechnen oft synonym verwendet. Dabei impliziert die Verteiltheit eine räumliche Trennung der Komponenten, die zur Gesamtlösung beitragen, gegebenenfalls auch über größere Distanzen.[66] Parallelität hingegen bezieht sich auf die Trennung in gleichzeitig ablaufende Komponenten, schränkt die räumliche Entfernung, in der Teile solcher Anwendungen ablaufen, jedoch ein. Beide Konzepte haben sich trotz einiger Gemeinsamkeiten aus verschiedenen Richtungen entwickelt und stehen grundsätzlich für unterschiedliche Ansätze.[67] Ihre Gemeinsamkeiten basieren zum einen auf der Nutzung mehrerer Prozessoren, die über ein Netzwerk verbunden sind und zum anderen in der zeitgleichen Abar-

[64] Vgl. www.kazaa.com

[65] Vgl. www.morpheus.com

[66] Vgl. Meister /Parallelisierungspotential/ 14

[67] Die folgenden Ausführungen wurden Vgl. Leopold /Computing/ 1, 3, 4 entnommen.

beitung und Kooperation der Aktivitäten und Prozesse. Ungeachtet dieser Gemeinsamkeiten unterscheiden sich die Ansätze: So gründet sich die Motivation des Parallelen Rechnen auf einem möglichen Geschwindigkeitsgewinn, während durch Verteiltes Rechnen die Ressourcennutzung verbessert werden soll. In vielen Anwendungen sind sowohl Geschwindigkeitsvorteile als auch eine optimale Ressourcennutzung wichtig, wobei beide Architekturen kombiniert eingesetzt werden können.

Leopold definiert Paralleles und Verteiltes Rechnen sehr kompakt und vorrangig auf den Ort der Berechnung bezogen: "Parallel computing splits an application up into tasks that are executed at the same time, whereas distributed computing splits an application up into tasks that are executed at different locations, using different resources."[68] Daraus kann der Eindruck entstehen, dass die Hardware beim Parallel Computing räumlich begrenzt angeordnet ist, während diese Begrenzung beim Distributed Computing aufgehoben ist und somit verstreut genutzt werden kann. Dies ist jedoch lediglich eine Tendenz und kann keinesfalls als ein überzeugendes Merkmal für eine strenge Trennung angesehen werden.[69]

Beim Parallelen Rechnen wird die Applikation in mehrere Teilaufgaben zerlegt und gleichzeitig bearbeitet.[70] Die Applikationen laufen in homogenen Umgebungen, häufig mit gemeinsamem Speicher, und werden als eigenständig betrachtet, mit dem Ziel, die Abarbeitungsgeschwindigkeit zu erhöhen. Beim Verteilten Rechnen hingegen nutzt der Rechner viele Ressourcen wie Drucker, Scanner u.a., die sich an verschiedenen physischen Orten befinden können und auf die der Rechner über eine so genannte verteilte Software zugreifen kann. Verteilte Systeme sind oft heterogen, offen (Vgl. Kapitel 3.1), dynamisch und haben keinen gemeinsamen physischen Speicher.

Die Konzepte des Parallelen sowie des Verteilten Rechnens beschäftigen sich mit verschiedenen Aspekten der Anwendung, wobei ein Verschmelzen paralleler und verteilter Systeme häufig eine gute Lösung darstellt.

Im Folgenden möchte ich einige Argumente gegen eine vollkommen eigenständige Lösung paralleler als auch verteilter Systeme aufführen. *Garg* diskutiert diese Thematik auf der Hardwareebene unter der Prämisse, dass die am meisten verbreiteten Modelle Mehrprozessorarbeitsplätze sind, die über ein Netzwerk verbunden werden.

---

[68] Leopold /Computing/ 3

[69] Vgl. Leopold /Paralleles und Verteiltes Rechnen/

[70] Die folgenden Ausführungen sind Leopold /Computing/ 3, 4 entnommen worden.

Verteilte Systeme sind aus technischer Sicht besser „skalierbar“ als parallele Systeme, denn der gemeinsame Speicher letzterer lässt eine Art Flaschenhals entstehen, sobald sich die Anzahl der beteiligten Prozessoren erhöht.[71] Darüber hinaus bieten verteilte Systeme eine höhere „Flexibilität“, die sich dadurch auszeichnet, dass einzelne Prozessoren auf einfache Art und Weise hinzugefügt oder entfernt werden können. Außerdem unterstützen verteilte Systeme die „Daten- und Ressourcenteilung“, die folglich von verschiedenen Organisationen genutzt werden können. Die „geographische Struktur“ von Anwendungen kann grundsätzlich als verteilt angesehen werden. Die niedrige Kommunikationsbandbreite kann dabei die lokale Verarbeitung, z.B. für drahtlose Netzwerke, verbessern. Verteilte Systeme sind „zuverlässiger“ als parallele Systeme, weil eventuell auftretende Fehler eines einzelnen Prozessors durch andere mit dem Netzwerk verbundene Prozessoren mühelos ausgeglichen werden können. Schließlich sind verteilte Systeme hinsichtlich „ökonomischer Aspekte“ vorteilhafter, da sich diese durch Breitbandnetzwerke und günstige Workstations realisieren lassen. Parallele Systeme hingegen bieten, aufgrund ihrer Verfügbarkeit an jedem Knoten im Netzwerk einen entscheidenden Vorteil. So können Updates auf Systemen mit einem gemeinsamen Speicher „schneller durchgeführt“ werden als bei Systemen mit verteilten Prozessoren, die über Nachrichten kommunizieren, insbesondere, wenn dabei mehrere Prozessoren beteiligt sind.

4.2 Übersicht über Parallele und Verteilte Architekturen

In der Literatur findet sich eine große Anzahl genutzter Modelle paralleler und verteilter Architekturen.[72] Eine sehr einfache aber deshalb nicht weniger bekannte Klassifikation der Rechnersysteme geht auf *Flynn* zurück. Er unterscheidet diese zunächst in vier Gruppen: SISD - „Single Instruction Stream, Single Data Stream“, SIMD - „Single Instuction Stream, Multiple Data Stream“, MISD - „Multiple Instruction Stream, Single Data Stream“ und MIMD - „Multiple Instruction Stream, Multiple Data Stream“. Die ersten beiden Buchstaben dieser Abkürzung geben Auskünfte über die Anzahl der Befehlsströme, während die letzten beiden Buchstaben über die Anzahl der Datenströme informieren.

---

[71] Die folgenden Ausführungen wurden Vgl. Garg /Elements/ 1-3 entnommen.

[72] Die folgenden Ausführungen sind Vgl. Stange /Innovative Rechnerarchitekturen/ und Vgl. Leopold /Computing/ 25 entnommen worden.

Im Kontext dieser Arbeit spielen SIMD[73] und MIMD[74] die Hauptrolle, wobei die Mehrheit heutiger Rechnersysteme in die Klasse der MIMD-Architekturen (Vgl. Abb. 4-1) eingeordnet werden kann.

Abb. 4-1 zeigt verschiedene Programmiermodelle paralleler und verteilter Architekturen im Überblick. Dabei werden die folgenden Klassen unterschieden: SIMD-Computer (SIMD), Symmetrische Multiprozessoren (SMPs), Cache-Coherent NUMA Architekturen (CC-NUMA), Parallelcomputer mit verteiltem Speicher, Cluster, Grids und lose gekoppelte verteilte Systeme.

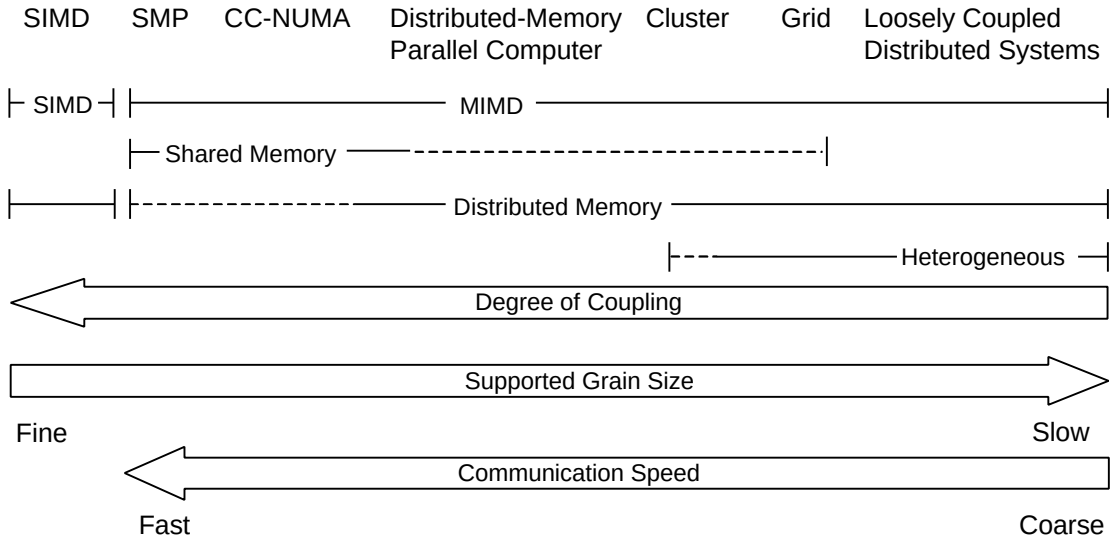

Abb. 4-1: Übersicht paralleler und verteilter Architekturen[75]

Die früheren parallelen Architekturen folgten oft dem Konzept der „SIMD-Computer“, die aus einem leistungsstarken Prozessor und einer Reihe relativ einfach konzipierter Verarbeitungseinheiten bestehen.[76] Der Speicher dieser Systeme ist verteilt angeordnet, so dass jedes Prozesselement über ein separates Speichermodul verfügt.

„Symmetrische Multiprozessoren“ sind seit Mitte der 60er Jahre im Einsatz und werden derzeit vorrangig als Server für kommerzielle Aktivitäten genutzt. Sie verfügen über

---

[73] Datenparallelität: Pipeline- und Feldrechner, Vgl. Stange /Innovative Rechnerarchitekturen/

[74] Programmparallelität: Homogene und inhomogene Multiprozessorsysteme, Vgl. Stange /Innovative Rechnerarchitekturen/

[75] Vgl. Leopold /Computing/ 49

[76] Die folgenden Ausführungen wurden Vgl. Meister /Parallelisierungspotential/ 14-16 und Vgl. Leopold /Computing/ 31-49 entnommen.

mehrere Prozessoren[77] und sind in Form einer „Bus- oder Switch-Architektur“ organisiert. Symmetrische Multiprozessoren haben einen einzelnen Adressraum, so dass jeder Prozessor jeden Speicherort über die Befehle „Laden“ und „Speichern“ erreichen kann. Die Prozessoren kommunizieren untereinander mittels schreiben zum bzw. lesen vom gemeinsamen Speicher.

Die „CC-NUMA[78] Architektur“ hat hinsichtlich des Adressraumes und der Kommunikation der Prozessoren ähnliche Eigenschaften wie die Symmetrischen Multiprozessoren. Der wesentliche Unterschied dieser Architekturen liegt in der Beziehung zwischen den Prozessoren und den Speichermodulen. Diese ist in SMPs symmetrisch, weshalb die Zugriffszeit auf eine Speicherzelle für alle Prozessoren gleich ist. In CC-NUMA Architekturen ist diese Relation unsymmetrisch, so dass jeder Prozessor über einen zugehörigen lokalen Speicher verfügt, auf den der Zugriff schneller erfolgen kann als auf den Restspeicher. Der Unterschied der Zugriffszeiten auf den lokalen Speicher und den Restspeicher vergrößert sich stetig bei steigender Anzahl der Prozessoren. Die Gründe dafür liegen zum einen in der Größe des Netzwerkverbundes und zum anderen am komplizierten Aufbau des „Cache Coherence“[79]-Protokolls der CC-NUMA Architektur gegenüber Symmetrischer Multiprozessoren.

Als „Parallelcomputer mit verteiltem Speicher“ werden parallele Architekturen mit keiner oder nur limitierter Unterstützung für den gemeinsamen Speicher bezeichnet. Die generische Struktur dieser ist identisch mit der von CC-NUMA Architekturen, allerdings entfällt die Hardwareunterstützung des Cache Coherence. Parallelcomputer mit verteiltem Speicher können zudem in NCC-NUMA[80] und NORMA[81] Architekturen unterschieden werden. Beide Systeme neigen dazu mehr Prozessoren als SMPs und CC-NUMA-Maschinen einzusetzen und sind darüber hinaus einfacher zu konstruieren, denn es wird keine Cache Coherence - Hardware benötigt.

---

[77] Gewöhnlich arbeiten SMPs mit 2 bis zu 64 Prozessoren.

[78] CC-NUMA - Cache Coherent Nonuniform Memory Access, Vgl. Leopold /Computing/ 37

[79] In einem Mehrprozessorsystem mit einem separaten Cache-Speicher für jeden Prozessor können viele Kopien in jedem Befehlsoperand vorliegen: eine Kopie im Hauptspeicher und eine in jedem Cache-Speicher. Sobald eine Kopie eines Operanden geändert wurde, müssen alle weiteren Kopien dieses Operanden ebenfalls geändert werden. „Cache Coherence“ ist die Methode um sicher zu stellen, dass Veränderungen der Werte der Operanden in zeitgemäßer Art und Weise im gesamten System zur Verfügung stehen. Vgl. o.V. /Cache Coherence/

[80] Non-Cache-Coherent Nonuniform Memory Access, Vgl. Leopold /Computing/ 37

[81] No Remote Memory Access, Vgl. Leopold /Computing/ 37

Als „Cluster“ versteht man ein Netz von „verteilten Arbeitsplatzrechnern oder Personal Computern, das über ein gemeinsames Programmiermodell mit Kommunikations- und Synchronisationsfunktionen für die Ausführung von parallelen Programmen geeignet“[82] ist. Cluster haben verschiedene Vorteile gegenüber konventionellen parallelen Architekturen, wobei der vergleichsweise niedrige Preis die wohl wichtigste Rolle spielt. Cluster können sowohl in die Klasse paralleler als auch verteilter Systeme eingeordnet werden.
„Lose gekoppelte verteilte Systeme“ sind ähnlich wie Cluster zu einem Netzwerk verknüpfte Arbeitsplatzrechner, PCs oder SMPs. Ein Unterschied zu Clustern besteht in dem vergleichsweise langsam verknüpften Netzwerk, welches gewöhnlich auf TCP/IP basiert. Aus diesem Grund werden diese gelegentlich für paralleles Rechnen eingesetzt, jedoch ist ihr Haupteinsatzfeld im verteilten Rechnen zu sehen. Die zu einem lose gekoppelten verteilten System verknüpften Komponenten sind oft heterogen, wobei die Gewährleistung der Sicherheit aufgrund der Größe der Netze nur schwer sicher zu stellen ist.
Die sich hinter „Grids“ verbergende Idee ist die Konzeption einer allgegenwärtigen Computerinfrastruktur, die dem elektronischen Stromnetz hinsichtlich deren Verfügbarkeit ähnelt. (Vgl. Kapitel 6) Bezüglich der Verbindung vieler Computerressourcen, Speicher und Datenbanken kommen sie Clustern am nächsten.
Das Spektrum der parallelen Architekturen, überwiegend auch als MIMD bezeichnet, reicht von SMPs über CC-NUMA, NCC-NUMA, NORMA bis hin zu Clustern, wobei die Grenzen dieser Systeme fließend sind. Die verschiedenen Architekturen unterscheiden sich dabei im Grad der Hardwareunterstützung der gemeinsamen Speicher sowie deren Kommunikation im Allgemeinen. Zu den verteilten Architekturen gehören, neben den in einer Doppelrolle auftretenden Clustern, Grids und lose gekoppelte verteilte Systeme.

5 Zentrale vs. Dezentrale Kommunikationsmodelle

Im folgenden Kapitel werde ich einen Überblick über Zentrale und Dezentrale Kommunikationsmodelle geben. Veranschaulichen möchte ich diesen am Beispiel Verteilter Betriebssysteme für Zentrale und am Beispiel des Peer-to-Peer (P2P) Konzeptes für Dezentrale Kommunikationsmodelle.

---

[82] Bode/ Cluster/ S. 139-140

5.1 Verteilte Betriebssysteme

Bei zentralen Kommunikationsmodellen übernehmen so genannte dedizierte[83] Server spezifische Aufgaben. Da in dezentralen Kommunikationsmodellen von vornherein eine gewisse Unsicherheit darüber herrscht, wer mit wem kommuniziert, sind alle Teilnehmer gleichberechtigt und somit in der Lage, die gleichen Dienste anzubieten.

5.1.1 Grundlagen und Funktionen von Betriebssystemen

Das Betriebssystem, in der Fachliteratur auch als „Operating System“ bezeichnet, „umfasst alle Systemprogramme, welche die Ausführung von Anwendungsprogrammen nach einer bestimmten Betriebsart steuern und alle dafür notwendigen Betriebsmittel sowohl verwalten als auch betreiben“[84]. Es teilt somit die Rechenleistung, den Speicher und die Geräte der Maschine zwischen allen Benutzern eines Rechners auf und verhindert die ungewollte Beeinflussung der verschiedenen Programme auf einem Rechner.[85] Für ein einzelnes Programm stellt das Betriebssystem eine Befehlsbibliothek dar, die die Maschinensprache eines Rechners um Funktionen wie „mehr Speicherplatz zur Verfügung stellen“, „Datensätze aus einer Datei lesen“ u.ä. erweitert. „Diese doppelte Aufgabenstruktur des Aufteilens und Multiplexens von knappen Betriebsmitteln auf der einen Seite und des Verbergens von lästigen Details aus der realen Welt vor den Anwendungen auf der anderen Seite bestimmt die Struktur eines Betriebssystems.“[86] Betriebsmittel, auch als „Resources“ bezeichnet „sind alle Hard- und Softwarekomponenten, die ein Programm braucht, um ausgeführt werden zu können. Zur Menge der Betriebsmittel zählen nicht nur alle Hardwarekomponenten wie Prozessor, Speicher und E/A-Geräte, sondern auch alle Softwareobjekte wie Programmvariable, Befehle, Daten und Dateien.“[87]

Im Bereich der Hardware ist der Prozessor die einzige aktive Komponente des Systems. In den meisten Betriebssystemen ist lediglich ein Prozessor vorhanden, der aus der

---

[83] In diesen Netzwerken ist der Server ausschließlich ein Server mit der Aufgabe, die Ressourcen wie Laufwerke, Drucker u.a. den Clients zur Verfügung zu stellen. Dieser Server wird als dedizierter Server bezeichnet, da er außer den Netzwerkdiensten keine weiteren Aufgaben ausführen kann.

[84] Wörn, Längle /Betriebssysteme/

[85] Köhntopp /Betriebssysteme/

[86] Köhntopp /Betriebssysteme/

[87] Wörn, Längle /Betriebssysteme/

abstrakten Sicht der Software als Kontrollfluss eines Programms erscheint.[88] Damit der Prozess ablaufen kann, muss er selbst und seine Daten gespeichert werden; dies übernimmt der Arbeitsspeicher als eine weitere Hardwarekomponente. Dieser hat die Aufgabe, zu verhindern, dass ein Prozess mehr als den ihm zugesicherten Speicher benutzt und dadurch möglicherweise andere Prozesse in ihrem Ablauf stört. Die Ein- und Ausgabegeräte werden zum Beispiel durch Drucker, Bandlaufgeräte etc. repräsentiert und sind zur Laufzeit nur von einem Prozess nutzbar. Dabei muss das Betriebssystem den Zugriff auf diese Geräte zuteilen, wie im Falle des Druckers über einen Druckerspooler. Andere Geräte wie Festplatten können aufgeteilt werden, indem das Betriebssystem einen entsprechenden Dienst zu Verfügung stellt.

Betriebsmittel können primär in reale, virtuelle und logische Betriebsmittel unterteilt werden. Während reale Betriebsmittel die reale Arbeitspeicherverwaltung, d.h. die reale Adressierung, die Unterbrechungsverwaltung sowie die Verwaltung der Prozessinteraktionen übernehmen, sind virtuelle Betriebsmittel für die virtuelle Arbeitsspeicherverwaltung, d.h. die virtuelle Adressierung, und die Prozessorverwaltung zuständig.[89] Logische Betriebsmittel sind für die Datei- und Programmverwaltung sowie für die Verwaltung von Ein- und Ausgabegeräten zuständig.

Das Betriebssystem der Zukunft wird mit den heute verbreiteten Systemen, die kaum mehr als dynamisch „linkbare“ Bibliotheken von Ein- und Ausgabefunktionen sind, wenig gemeinsam haben.[90] Ähnlich kann es experimentellen Systemen wie „Mach“[91], „Plan9“[92] und „Amoeba“[93] ergehen, die nach dem Client-Server-Modell aufgebaut sind. Dabei sendet der Aufrufer dem Server eine Nachricht, die von diesem ausgewertet wird und dem Absender als Resultat in Form einer Antwortnachricht zugestellt wird.[94] Der Server ist kein Teil des Betriebssystems selbst, sondern ein normaler Prozess, der möglicherweise einige zusätzliche Privilegien gegenüber gewöhnlichen Anwendungen, wie die Zugriffsmöglichkeit auf einige Hardwareregister, hat. „Es wird einen kleinen Kern mit den notwendigsten Funktionen zum „Multitasking“ und „Message Passing“ haben

---

[88] Die nachfolgenden Ausführungen sind Vgl. Köhntopp /Betriebssysteme/ entnommen worden.

[89] Vgl. Wörn, Längle /Betriebssysteme/

[90] Vgl. Köhntopp /Betriebssysteme/

[91] Vgl. www-2.cs.cmu.edu/afs/cs.cmu.edu/project/mach/public/www/mach

[92] Vgl. www.cs.bell-labs.com/plan9dist/

[93] Vgl. www.cs.vu.nl/pub/amoeba/

[94] Vgl. Köhntopp /Betriebssysteme/

und eine große Zahl von Standartserverprozessen für die unterschiedlichen Systemdienste bereitstellen.[95] Zu diesen Systemdiensten wird das Betreiben von Geräteabstraktionen wie Standarddruckern und einheitlich steuerbaren Grafikbildschirmen ebenso gehören wie das Bereitstellen komplexer Bibliotheksfunktionen über Serverprozesse mit eigenem Kontrollfluss für die Konvertierung von Datenformaten.[96]

5.1.2 Definition und Klassifizierung verteilter Betriebssysteme

„Der Grad der Vernetzung ist drastisch angestiegen. Das weitgehende Fehlen adäquater Hilfsmittel für die Steuerung vernetzter Umgebungen lässt nun die Anwendung von Betriebssystemen, die auf eine einzelne Maschine beschränkt sind, fraglich erscheinen. Gefordert sind jetzt verteilte Betriebssysteme (VBS), die Netze und angeschlossene Rechner als logische Einheit steuern können.“[97] In der Fachliteratur wird grundsätzlich zwischen zentralen und verteiltem Betriebssystem unterschieden. „Man spricht von einem zentralen Betriebssystem, wenn SISD-, SIMD- oder eng gekoppelte MIMD-Rechner (Mehrprozessorsysteme) damit gesteuert und überwacht werden.“[98]

Ein verteiltes Betriebssystem, auch als „Distributed Operating System“ bezeichnet, „ist ein Betriebssystem, bei dem alle Funktionseinheiten auf ein verteiltes Rechnersystem so aufgeteilt werden, daß durch die Interaktion zwischen diesen verteilten Einheiten“[99], die in einfachen Betriebssystemen notwendigen Funktionen (Vgl. Kapitel 5.1.1), sowohl ortstransparent als auch skalierbar erbracht werden.

Verteilte Betriebssysteme können in folgende Klassen unterteilt werden: verteilte Betriebssystem-Kerne (VBS-Kerne) mit minimaler Funktionalität, integrierte Systeme, objektorientierte Systeme und Systeme, die auf einem Server-Pool-Modell basieren. Diese Einteilung schließt jedoch nicht aus, dass ein System mehreren Klassen angehört; so „kann beispielsweise ein System auf einem Server-Pool-Modell beruhen und dennoch objektorientiert sein“[100].

VBS-Kerne bieten eine fast minimale Anzahl von Funktionen für das Prozessmanagement, die Speicherverwaltung, den Nachrichtenaustausch und die Verwaltung von Hin-

---

[95] Köhntopp /Betriebssysteme/

[96] Vgl. Köhntopp /Betriebssysteme/

[97] o.V. /Verteilte Betriebssysteme/

[98] Wörn, Längle /Betriebssysteme/

[99] Wörn, Längle /Betriebssysteme/

[100] o.V. /Verteilte Betriebssysteme/

tergrundspeicher-Medien an. Beispiele für diese Klasse verteilter Betriebssysteme sind „Accent“[101] und „Mach“ von der Carnegie Mellon Universität sowie der „V-Kernel“[102] der Universität Standford.

„Integrierte Systeme ermöglichen es, auf jeder Maschine eine in weiten Grenzen konfigurierbare Standardsoftware laufen zu lassen, die die Maschine mit allen benötigten Hilfsmitteln versorgt. Diese Umgebungen entstanden im Umfeld der Betriebssystem-Entwicklung von Unix und „stellen daher teilweise die bisherigen Unix-Funktionen zur Verfügung, so daß eine gewisse Kompatibilität zu älteren Programmen gewährleistet ist“[103]. Als Beispiele hierfür können „Locus“[104] von der Universität von Kalifornien entwickelt und der Locus Incorporation geführt sowie das „Cambridge Distributed Operating System“[105] und „Saguaro“[106] von der Universität von Arizona genannt werden.[107]

„Objektorientierte Systeme halten für die Anwendungen die Sichtweise der abstrakten Objekte und der auf ihnen durchzuführenden ebenfalls abstrahierenden Operationen streng durch.“[108] Diese Eigenschaften erfüllen unter anderem das System „Argus“[109] des Technischen Instituts von Massachusetts, „Eden“[110] der Universität von Washington sowie „Amoeba“ der Universität von Amsterdam.

„Systeme die auf einem Server Pool Modell beruhen, stellen dem Benutzer je einen Verbund von Servern für die angestrebten Services zur Verfügung. Ein einfaches System dieser Klasse ist das Cambridge Distributed Operating System.“[111]

5.1.3 Vorteile und Nachteile verteilter Betriebssysteme

Zu den Vorteilen verteilter Betriebssysteme gegenüber Einzelsystemen gehören ein verbessertes Preis-Leistungsverhältnis, die Steigerung der maximalen Leistungsfähigkeit durch die Parallelverarbeitung, die erhöhte Verfügbarkeit durch unabhängiges Agieren

---

[101] Vgl. Rashid, Robertson /Accent/

[102] Vgl. www.cs.wisc.edu/~sschang/OS-Qual/distOS/Vkernel.htm

[103] o.V. /Verteilte Betriebssysteme/

[104] Vgl. Sheth /Locus/

[105] Vgl. http://research.microsoft.com/camdis

[106] Vgl. Andrews et al. /Saguaro/

[107] Vgl. o.V. /Verteilte Betriebssysteme/

[108] o.V. /Verteilte Betriebssysteme/

[109] Vgl. Shrivastava, Dixon, Parrington /Reliable Distributed Systems/

[110] Vgl. Almes et al. /Eden/

[111] o.V. /Verteilte Betriebssysteme/

der Komponenten und die Skalierbarkeit.[112] Aus ökonomischer Sicht ist es erfahrungsgemäß von Vorteil, einfache und damit kostengünstige Hardware zu bevorzugen als für einzelne Anwendungsfelder Spezialhardware zu beschaffen. Darüber hinaus können sowohl die Teilung von Daten (z.B. verteilte Datenbanken) als auch von Geräten (z.B. Drucker-Server) und die Teilung der Belastung, aufgrund der Übernahme von Aufgaben durch freie Rechner, als wesentliche Erfolge verteilter Betriebssysteme angesehen werden.

Als Nachteile verteilter Betriebssysteme sind einerseits die sehr schwierige Datensicherheit und andererseits die Fehleranfälligkeit der Netzwerke anzusehen. Weiterhin kann das Management verteilter Betriebssysteme zum Beispiel in Unix-Netzen problematisch werden und die erhöhte Kommunikation für die Parallelisierung viel Leistung verbrauchen.

5.2 Peer-to-Peer-Netzwerke (P2P)

„Der Begriff Peer-to-Peer unterliegt einer Inflation, wie sie bei interessanten Technologien häufig auftritt“[113] und „hat sich zu einem der meistdiskutierten Begriffe der Informationstechnologie herausgebildet“[114]. Darunter werden dezentrale Systeme verstanden, die aus lose gekoppelten Knoten mit gleichem Funktionsumfang bestehen.[115]

Demzufolge können alle Knoten des Peer-to-Peer-Systems die gleichen Funktionen erfüllen. Auf der Organisationsebene handelt es sich deshalb bei dem P2P-Modell „um das Gegenteil des Client-Server-Modells, da es in einzelnen Ansätzen lose strukturierte, grundsätzlich jedoch dezentrale Systeme umfasst“[116]. Attraktive Anwendungsbeispiele für die P2P-Technologie sind die „Beschleunigung von Kommunikationsprozessen, die hohe Austauschfähigkeit aktueller, dezentral generierter Informationen, die bessere Auslastung ´brachliegender´ Ressourcen oder die effiziente Bewirtschaftung von knappen Ressourcen“[117].

Trotz einiger Vorteile und Vereinfachungen wie z.B. fallende Kosten, steigende Verfügbarkeit von Rechenleistung, Bandbreite und Speicherplatz wirft der Verzicht einer

---

[112] Die nachfolgenden Ausführungen sind Vgl. Böszörmenyi /Betriebssysteme/ entnommen worden.

[113] Strufe /Peer-to-Peer-Distributionssystem/

[114] Schoder, Fischbach /Ressourcenmanagement/ 313

[115] Vgl. Strufe /Peer-to-Peer-Distributionssystem/

[116] Strufe /Peer-to-Peer-Distributionssystem/

[117] Schoder, Weinhardt /Technologien/ 257

zentralen Koordination bedeutende Fragen auf: „Wie kann eine dezentrale Kontrolle mit den Herausforderungen hinsichtlich dauerhafter Verfügbarkeit, Sicherheit und fairer Kostenaufteilung umgehen? Welche Dienste müssen geeignete Infrastrukturen für P2P-Applikationen erbringen?“[118]

5.2.1 Begriffsklärung und Konzept von P2P-Netzwerken

In der Literatur gilt die P2P-Technologie seit den Erfolgen der Musiktauschbörsen wie „Napster“[119] und „Gnutella“[120], um nur einige zu nennen, als ein Trend für die Zukunft des Internet.[121] Die Grundlage der Peer-to-Peer-Netzwerke liegt in der Theorie zu Verteilten Systemen. (Vgl. Kapitel 3) Das Wort „Peer“ bedeutet gleichrangig beziehungsweise gleichgestellt und bezeichnet alle Geräte in einem geschichteten Kommunikationsnetzwerk, die auf der gleichen Protokollebene arbeiten.[122] Demzufolge bedeutet „Peer-to-Peer“ soviel wie von gleich zu gleich, da alle Geräte in der Lage sind die gleichen Dienste auf der gleichen Protokollebene auszuführen. Eine einheitliche Definition des sich hinter dem Begriff P2P-Netzwerk verbergenden neuartigen Konzepts kann selbst von der Fachliteratur bislang nicht vorgenommen werden. Während *Tanenbaum* und *van Steen* diesen Begriff als eine Architekturvariante ohne Server verstehen, bei dem die Client-Applikationen direkt miteinander kommunizieren, definiert *Webopedia* diesen als ein Netzwerk, in dem jede Workstation die gleichen Fähigkeiten und Kompetenzen hat.[123] Beide Definitionen heben insbesondere die Unabhängigkeit der Server als auch die Gleichrangigkeit der Knoten im Netzwerk hervor.[124] Dies wird nachfolgend als Peer-to-Peer im engeren Sinne verstanden.

Darüber hinaus gibt es P2P-basierte Anwendungen, bei denen Server eingesetzt werden und somit nicht der Definition des P2P im engeren Sinne entsprechen.[125] Die *peer-to-peer-working group* beschreibt P2P in einer weiter gefassten Definition als das Teilen

---

[118] Schoder, Weinhardt /Technologien/ 257

[119] Vgl. www.napster.com

[120] Vgl. www.gnutella.com

[121] Vgl. Freismuth /P2P-Zukunftsmodell/

[122] Vgl. o.V. /P2P-Begriffe/, Vgl. Frascaria /Peer-to-Peer/

[123] Vgl. Tanenbaum, van Steen /Distributed Systems/ 53, Vgl. o.V. /Peer-to-Peer/

[124] Vgl. Heinzl /Peer-to-Peer-Netzwerk/

[125] Vgl. Frascaria /Peer-to-Peer/

von Ressourcen und Diensten durch den direkten Austausch zwischen den Systemen.[126] Dabei werden für den direkten Austausch zwischen den Knoten Server zur Unterstützung des Verbindungsaufbaus eingesetzt. Demzufolge wird die Definition der *peer-to-peer-working group* als Peer-to-Peer im weiteren Sinne verstanden.

5.2.2 Konzept und Dreiebenenmodell von P2P-Netzwerken

Charakteristisch für P2P-Netzwerke sind neben der „gegenseitigen Bereitstellung von Ressourcen“ außerdem „Dezentralität“ und „Autonomie“.[127] Die gegenseitige Bereitstellung von Ressourcen bedeutet, dass in einem solchen Netzwerk jeder Knoten sowohl Client- als auch Serverfunktionalität leisten kann und somit Anbieter und auch Nachfrager von Diensten und Ressourcen sein kann. Das Merkmal der Dezentralität impliziert das Fehlen einer zentralen Koordinationsinstanz für die Organisation des Netzwerkes und die Kommunikation zwischen den einzelnen Peers des Netzes. Folglich kann kein Knoten einen anderen zentral kontrollieren. Die Kommunikation zwischen den Peers erfolgt direkt. Durch die Autonomie der Knoten in einem P2P-Netzwerk, können diese mehr oder weniger selbst bestimmen, wann und in welchem Umfang sie Ressourcen zur Verfügung stellen.

Die in Theorie und Praxis zuweilen unscharfe Begriffsverwendung (Vgl. Kapitel 5.2.1) von P2P-Netzwerken kann in einem Dreiebenenmodell weitestgehend aufgelöst werden. Die Ebene 1 repräsentiert P2P-Infrastrukturen und liegt oberhalb von Telekommunikationsnetzwerken, die als Basis für alle darüber liegenden Ebenen fungieren. Sie leisten Kommunikations-, Integrations- und Übersetzungsfunktionen zwischen IT-Komponenten und stellen Dienste zur Verfügung, um Peers im Netzwerk zu finden, mit ihnen zu kommunizieren, Ressourcen zu identifizieren, zu nutzen und auszutauschen. Die Ebene 2 wird aus P2P-Anwendungen gebildet, die Dienste der darunter liegenden P2P-Infrastrukturen nutzen und die Kommunikation und Kooperation ohne Koordinationsinstanzen sicherstellen.

Auf der Ebene 3 erfolgt die Bildung von P2P-Gemeinschaften durch die kooperative Interaktion gleich gesinnter oder gleich gestellter Personen. „Im Gegensatz zu den Ebenen 1 und 2, wo der Begriff Peer grundlegend alle technischen Instanzen bezeichnet, die auf

---

[126] Vgl. o.V. /Peer-to-Peer working group/

[127] Die folgenden Ausführungen sind Vgl. Schoder, Fischbach /Ressourcenmanagement/ 313, 314 entnommen worden.

gleicher Ebene arbeiten, wird auf Ebene 3 die Bedeutung des Begriffs Peer nichttechnisch interpretiert“[128], sondern „Peer als Person“ gedeutet.

5.2.3 Anwendungsmöglichkeiten von P2P-Netzwerken

Das Hauptziel verteilter Systeme besteht im Teilen von Ressourcen zwischen Nutzern.[129] Demzufolge besteht das Ziel von lokalen P2P-Netzen zum einen in der Nutzung bereits verfügbarer Ressourcen und zum anderen in der Bereitstellung eigener.[130] Die in P2P-Netzwerken zu beachtenden Ressourcentypen sind Information, Dateien, Bandbreite, Speicherplatz und Rechnerleistung.[131]

Einer der bedeutendsten Anwendungsbereiche von P2P-Netzwerken im Internet ist der Austausch von Dateien, das so genannte File Sharing.[132] Bezeichnend dafür ist, dass zuvor als Client agierende Peers, Dateien oder deren Fragmente herunter geladen haben, unmittelbar danach als Server auftreten und diese anderen zur Verfügung stellen.

Darüber hinaus besteht die Möglichkeit der Bereitstellung von Speicherplatz innerhalb des P2P-Netzwerkes. Der Teilnehmer an einem P2P-Speichernetzwerk erhält für jeden Peer einen privaten und einen öffentlichen Schlüssel, mit dessen Hilfe mittels einer Hash-Funktion jeder Peer eindeutig identifiziert werden kann. Speicherplatz auf anderen Rechnern muss im Austausch mit eigenem Speicherplatz bzw. über die Entrichtung einer Gebühr erworben werden.[133]

Ein weiterer Anwendungsbereich von P2P-Netzwerken wird in der Kollaboration gesehen.[134] Zu diesem Bereich können Instant Messaging-, Chat-Anwendungen und Dokumentenbearbeitung gezählt werden. Somit erfolgt die Kommunikation zwischen den Teilnehmern des Netzwerkes auf direktem Weg. Der Anwendungsbereich der Kollaboration hat seine größte Bedeutung in der unternehmensinternen Nutzung. Die bekannteste Anwendung in diesem Zusammenhang ist die „Groove Workspace“. „Diese bietet den Nutzern die Möglichkeit, selbst so genannte „shared spaces“ einzurichten, die je-

---

[128] Schoder, Fischbach /Ressourcenmanagement/ 314

[129] Vgl. Tanenbaum, van Steen /Distributed Systems/ 4

[130] Vgl. Heinzl /Peer-to-Peer-Netzwerk/

[131] Vgl. Schoder, Fischbach /Ressourcenmanagement/ 316

[132] Vgl. Schoder, Fischbach /Ressourcenmanagement/ 317

[133] Vgl. Schoder, Fischbach /Ressourcenmanagement/ 318; Vgl. www.hivecache.com

[134] Vgl. Lange /Alternativen/

weils eine gemeinsame Arbeitsumgebung für ad hoc gebildete virtuelle Teams bilden sowie andere Nutzer zur Zusammenarbeit in diesen Teams einzuladen.“[135]

Neben den bereits aufgeführten Anwendungsmöglichkeiten existieren unzählige weitere Einsatzgebiete von P2P-Netzwerken. Ein bis zum heutigen Zeitpunkt vergleichsweise kaum genutzter Ansatz ist der Aufbau dezentral organisierter elektronischer Marktplätze. Während durch den Wegfall eines zentralen Marktplatzbetreibers die Teilnahme ohne Gebühren erfolgt, ist die Realisierung von Zusatzdiensten wie Versand, Bewertung von Marktteilnehmern u.a. sehr problematisch.

6 Das Grid

Im folgenden Kapitel soll das Konzept der Grid-Technologie beleuchtet werden. Beginnend mit einer Einführung in das Thema und der Begriffsklärung des Grid werde ich dessen Architektur sowie Anwendungsgebiete und Visionen von Grids erläutern.

6.1 Einführung in die Grid-Technologie

Der Begriff „Management by Delegation“, als Spezialisierung des Managementbegriffes, beschreibt die Führung durch Aufgabenübertragung an untergeordnete Hierarchieebenen. Durch die zunehmende Arbeitsteilung und Spezialisierung ist die Delegation von Verantwortlichkeiten und Kompetenzen zur Entlastung der Führungsebene beinahe zwangsläufig.

Dieses „Management by Delegation“, welches in betriebswirtschaftlicher Hinsicht die Kunst Arbeit sinnvoll zu verteilen beschreibt, soll durch neue Technologien auch auf das Netz übertragen werden.[136] Dazu verbinden so genannte Grids „ansonsten teilweise brachliegende Ressourcen für sinnvolle Aufgaben.“[137]

Diese neue Netzstruktur soll den Umgang mit komplexen Problemen vereinfachen, um Ressourcen zu verteilen oder zu bündeln, gemeinsam an Daten zu arbeiten oder Netzdienste anzubieten.[138] Diese Sachverhalte existieren zwar schon weitestgehend einzeln, jedoch bringt das Konzept der Grid-Technologie einerseits die verschiedenen Technologien zusammen und andererseits in entscheidenden Details wesentliche Verbesserun-

---

[135] Schoder, Fischbach /Ressourcenmanagement/ 316

[136] Vgl. Gleich /Intergrid/ 208

[137] Gleich /Intergrid/ 208

[138] Die Ausführungen des folgenden Abschnitts sind Vgl. Gleich /Intergrid/ 208 entnommen worden.

gen. Zur Verdeutlichung kann das Projekt „SETI@home“, die Suche nach extraterrestrischen Funksignalen dienen, welche Anzeichen einer intelligenten Absicht vorweisen können. Trotz dieser gegenwärtig herrschenden Euphorie treten auch Probleme auf, die es zu bewältigen gilt:

Der Nutzer lädt sich ein Programm auf seinen Rechner, von dem er nicht so genau weiß, was dieses eigentlich genau macht. Auf diese Art weisen die Server der Forschungsteams jedem ihrer Projektteilnehmer Datenblöcke zu, die die Teilnehmer bearbeiten sollen. Dabei ist es nahezu unmöglich, jedem der Teilnehmer gleichviel Rechenzeit zu überlassen, da jeder Nutzer seinen individuell zur Verfügung stehenden Zeitrahmen hat; die Bearbeitung der eigenen Aufgaben steht im Vordergrund.

An diesem Punkt setzt die eigentliche Idee an: Das Grid soll eine einheitliche Protokollebene schaffen, deren Implementierung Aufgabe des Betriebssystems und der Anwendungen ist.[139] Dadurch könnte Rechenzeit portioniert all den Anwendungen zur Verfügung gestellt werden, deren Ziele vom potentiellen Nutzer als sicher einschätzt werden. Dies könnte so erfolgen, dass der Nutzer festlegt, wie viel Prozent seiner Rechenleistung er selbst für seine aktuelle Anwendung benötigt und wie viel er von den übrigen Ressourcen z.B. Forschungsinitiativen überlässt. Die dafür benötigte Infrastruktur stellt das Grid bereit, wobei jedes der Projekte von jedem als Teilnehmer markierten Nutzer die Rechenzeit anfordert.

6.2 Begriffsklärung des Grid

Der Begriff des Grid, treffend übersetzt durch „Gitter“, wurde Mitte der Neunziger Jahre geprägt, um eine koordinierte Computer-Infrastruktur zu benennen, in der Ressourcen an jedem Knotenpunkt des Netzes verfügbar sind[140]. Es soll sich um die Verteilung der Lasten kümmern.[141] Vereinfachend lässt sich das Grid als „eine Kommunikationsstruktur beschreiben, in der transparent und einheitlich auf verschiedene Ressourcen zugegriffen werden kann, ohne für jedes Problem die dazugehörige Netzstruktur neu erfinden zu müssen.“[142] Es ist eine Sammlung verteilter „Computing Ressourcen“, welche über Netze wie Wide Area Network (WAN) und Local Area Network (LAN) verbunden sind und dem Endbenutzer bzw. einer Applikation wie ein einzelnes großes virtuelles

[139] Die Ausführungen des folgenden Abschnitts sind Vgl. Gleich /Intergrid/ 208, 209 entnommen worden.

[140] Vgl. Gleich /Intergrid/ 209

[141] Vgl. Gleich /Intergrid/ 209

[142] Gleich /Intergrid/ 208

System erscheint.[143] Dabei wird über die Grenzen von einzelnen Arbeitsplätzen, Einrichtungen oder Ressourcen hinaus eine Zusammenarbeit durch standardisierte, sichere und koordinierte Ressourcennutzung möglich. Das Grid Computing erlaubt das so genannte Verteilte Rechnen nicht nur lokal, sondern auch über Einrichtungs-, System- und Software-Grenzen hinweg und erleichtert dadurch jedem, der innerhalb eines solchen Grids ist, die Nutzung der Rechenleistung, Daten oder Informationen der anderen.

Der Begriff des Grid wird in der Fachliteratur unterschiedlich definiert. Auch wenn die nachfolgenden Definitionen dem Sinn nach ähnlich sind, unterscheiden sie sich doch in ihrem Detailliertheitsgrad. Während *Mariske* Grids lediglich als „Zusammenschaltung von Computern in neuen, leistungsfähigen Netzwerken“[144] definiert, beschreibt *Klaß* diese als „beständige Umgebungen, in denen Anwendungen die Prozessorleistung und Speicherkapazität von ganzen Computer-Clustern gemeinsam über das Internet nutzen“[145].

Ich möchte meine Ausführungen zum Konzept des Grid, an der Definition von *Foster* und *Kesselmann* anlehnen, welche den Begriff eingeführt haben. „A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.“[146] Im Folgenden möchte ich auf die einzelnen Bestandteile dieser Definition näher eingehen.

Im Fokus der Grid-Technologie steht die verteilte Nutzung von Ressourcen, wie beispielsweise Speicher- und Rechnerleistung bzw. die Verbindung von Daten und Computer-Clustern.[147] Diese Verteilung setzt eine signifikante „Hardware-Infrastruktur“ voraus, welche die Realisierung der notwendigen Verbindung ermöglicht. Darüber hinaus ist eine „Software-Infrastruktur“ zur Darstellung und Kontrolle des resultierenden Verbunds notwendig.

Eine grundlegende Eigenschaft des Grids stellt das Element „dependable“ dar. Darunter wird die Zusicherung einer vorhersehbaren und ununterbrochenen Performance der verschiedenen Teilkomponenten des Grids auf höchster Ebene verstanden. Die Eigenschaften der Performance variieren von Anwendung zu Anwendung, werden aber insbeson-

---

[143] Die folgenden Ausführungen sind Vgl. o.V. /Grid Computing entnommen worden.

[144] Mariske /World Wide Grid/

[145] Klaß /Kommerzielles Grid-Computing/

[146] Foster, Kesselmann /Computational Grids/ 3

[147] Die folgenden Ausführungen sind Vgl. Foster, Kesselmann /Computational Grids/ 3 entnommen worden.

dere durch die vorhandene Bandbreite, eventuelle Schwankungen und Wartezeiten, Rechenleistung und Softwarequalität sowie Sicherheit und Zuverlässigkeit bestimmt.
Der Bestandteil „consistent“ stellt ein weiteres grundlegendes Element dar und beschreibt die Qualität der Services für die Anwender, unabhängig vom Zeitpunkt, vom Standort und von der Hardware über welche die Verbindung hergestellt wird. Daraus resultiert die Forderung nach Standardisierung, welche sowohl standardisierte Schnittstellen als auch den Betrieb über standardisierte Parameter einschließt.
„Pervasive“ fokussiert auf den Bedarf der Anwender, ähnlich wie bei der Stromversorgung, zu jeder Zeit von jedem Standort aus auf die Funktion des Grid zugreifen zu können und impliziert damit die Anforderung nach Verlässlichkeit und Garantie der dauernden Verfügbarkeit dieser Leistung.
Die Eigenschaft „inexpensive“ zielt auf die Nutzbarkeit der Grid-Technologie in ökonomischer Hinsicht ab. Um diesem Konzept in naher Zukunft eine erhöhte Akzeptanz zu ermöglichen, besteht der Anspruch nach günstigen Zugangs- und Nutzungskosten.
„Im Netz gibt es einige Gruppierungen, die sich mit der Entwicklung der nötigen Software und Ideen auseinandersetzen“[148], wobei der von der Globus-Gruppe entwickelte Ansatz, der „Globus-Grid“, von Experten als kommender Standard im Grid-Computing prognostiziert wird.[149] In diesem „Globus-Grid“ wird ein Anbieter von Diensten, unabhängig von deren Art, als Virtuelle Organisation, kurz „VO“, bezeichnet.[150] Eine „VO“ wird als eine Ansammlung von Individuen und/oder Organisationen mit geographisch und organisatorisch verteilten Komponenten virtueller Computersysteme verstanden.[151] Dabei muss eine solche Organisation nicht zwangsläufig geografisch zusammenhängen, sondern könnte ebenso „aus einem Zusammenschluss örtlich weit entfernter Firmen oder Individuen mit gemeinsamen Zugriffsregeln bestehen“[152]. Eine weitere erforderliche Eigenschaft einer „VO“ ist das Vorhandensein von Regeln zur Organisation der Verteilung und Verfahrensweisen für das Agieren der Teilnehmer, wobei die Grid-Technologie die verteilte Nutzung sowie Koordination der verschiedenen Ressourcen unterstützen soll.[153]

---

[148] Gleich /Intergrid/ 209

[149] Vgl. Gleich /Intergrid/ 209

[150] Vgl. Gleich /Intergrid/ 209

[151] Vgl. Foster, Kesselmann, Tücke /Anatomy of the Grid/ 2, vgl. Foster et al. /Physiology of the Grid/ 5

[152] Gleich /Intergrid/ 209

[153] Vgl. Foster, Kesselmann, Tücke /Anatomy of the Grid/ 2

6.3 Architektur von Grids

Ein wichtiges Element bei der Gestaltung einer effizienten Organisation ist die Interoperabilität und Kompatibilität, welche durch den Einsatz gemeinsamer Protokolle gewährleistet wird. In seiner jetzigen Form baut das Netz auf die vorhandene Infrastruktur auf, jedoch könnten in naher Zukunft neue Protokolle und Sicherheitsmechanismen für das Grid erforderlich werden.[154] Gegenwärtig werden die verfügbaren Sicherheitslösungen und Verbindungsarten genutzt sowie die Internet Protokolle (IP) einschließlich deren aufsetzenden Transportmechanismen, wie beispielsweise Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP) oder Lightweight Directory Access Protocol (LDAP). Diese Protokolle sind in Schichten angeordnet und folgen dem so genannten Stundenglasansatz, welcher in Abb. 6-1 grafisch dargestellt ist.

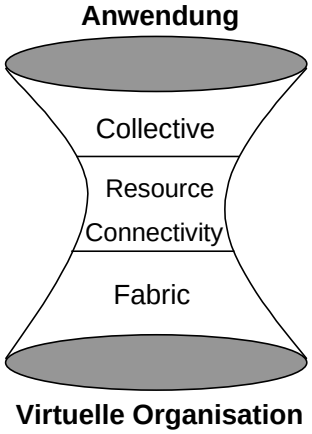

Abb. 6-1: Stundenglasansatz der Globus-Gruppe[155]

„Dieser besagt, dass einfache Kernschichten (der Flaschenhals) nur die nötigsten Funktionen beinhalten, während so viel wie möglich an den breiten Enden der Sanduhr stattfindet.“[156] In dem bereits erwähnten Ansatz der Globus-Gruppe (Vgl. Kapitel 6.2) sind am engen Flaschenhals des Glases die Schichten „Resource“ und „Connectivity“ zu finden.[157] Diese sollen schlank und möglichst einfach sein, aber gleichzeitig die grundlegenden Funktionen der Teilung von Ressourcen bieten können ohne die breiten Enden des Glases zu stark einzuschränken. Im Wesentlichen beschreiben sie einfache und si-

[154] Die folgenden Ausführungen sind Vgl. Gleich /Intergrid/ 209 entnommen worden.

[155] Vgl. Gleich /Intergrid/ 209

[156] Gleich /Intergrid/ 209

[157] Die folgenden Ausführungen sind Vgl. Gleich /Intergrid/ 209 entnommen worden.

chere Kommunikationsmechanismen und Methoden zum Zugriff auf einzelne Ressourcen. Die bei der Koordination der verteilten Mittel anfallenden Verwaltungsanforderungen werden durch die „Collective“-Schicht übernommen, auf welche Grid-fertige Anwendungen zurückgreifen können. Dabei werden die Ressourcen und die dazu gehörigen Verbindungen zu komplexen Funktionen gesammelt und der darüber liegenden Anwendungsebene zur Verfügung gestellt.

Am anderen Standfuß des metaphorischen Stundenglases liegt die so genannte „Fabric“-Schicht, wörtlich übersetzt Gewebe, und präsentiert die Serverseite. In dieser Schicht sind die Schnittstellen zu den internen Protokollen enthalten, die neben Plattformunabhängigkeit auch lokale Kontrolle garantieren sollen. Eine Virtuelle Organisation kann Daten, Kataloge oder Echtzeitmesswerte an jeden Nutzer mit anderen Rechten bereitstellen. Diese dürfen bereits Berechnungen auf dem Grid ausführen, das sich Daten von anderen Rechnern holt. Dazu werden zuvor von der jeweiligen Infrastruktur die Informationen über die für sie vorhandenen Mittel eingeholt.

6.4 Anwendungsgebiete und Visionen von Grids

Das Hauptziel des Einsatzes von Grid-Systemen ist die Unterstützung von Forschungstätigkeiten durch ein neuartiges Konzept der wissenschaftlichen Zusammenarbeit. So soll es den Wissenschaftlern ermöglicht werden, Prozessorleistung, Speicherkapazität und Anwendungen gemeinsam über das Internet zu nutzen, um Forschungsprojekte aus den Bereichen Biotechnologie, Klimaforschung u.a. voranzutreiben.[158] Mit Hilfe dieses Hochleistungsnetzwerks können Forscher zukünftig „Daten anderer Institute abfragen und auswerten, Anwendungen auf fremden Computern laufen lassen und komplexe Computersimulationen in Echtzeit auf weit voneinander entfernten Computern ansehen“[159]. Darüber hinaus werden Grids im Bereich kommerzieller Anwendungen z.B. bei Banken und Versicherungen für komplexe Risikoanalysen,[160] oder im Online-Spiele-Markt eingesetzt. Mit letzterem beschäftigt sich das Projekt „Butterfly Grid“ [161], welches Anbietern von Online-Spielen ermöglichen soll, einer Vielzahl von Nutzern den Zugang zu einem einzigen Spiel zu gewährleisten.[162]

---

[158] Vgl. Ihlenfeld /IBM - Größtes Grid/

[159] Ihlenfeld /IBM - Größtes Grid/

[160] Vgl. Herrmann /Grid Computing/ 43

[161]Vgl. www.butterfly.net

[162] Vgl. Ihlenfeld /Butterfly/

Die Vision der Grid-Technologie lässt sich als „Computing power may one day be as ubiquitous and easy to access as electricity“[163] beschreiben. Demzufolge soll die Abarbeitung eines komplexen Problems, welches auf dem Rechner des Nutzers gestartet wird, auf anderen Rechnern überall auf der Welt ausgeführt werden, ohne dass es für den Nutzer von Bedeutung ist, wo die Berechnung stattfindet und welcher Teilnehmer die Ressourcen dafür bereitstellt.[164]

7 Das Konzept des Metacomputing

In diesem Kapitel wird der Begriff des „Metacomputing“ sowohl im Allgemeinen als auch im Speziellen beleuchtet und in den Kontext des Parallelen und Verteilten Rechnens eingeordnet.

7.1 Metacomputing im Allgemeinen

In den vergangenen Jahren sind mit der Entstehung komplexer Anwendungen und Problemstellungen auch die Anforderungen an die Rechenleistung nahezu proportional gestiegen.[165] Da diese häufig nicht mehr von einem Einzelplatzrechner erfüllt werden können, wird der Trend zum Einsatz von verteilten Systemen sichtbar. Unter einem verteilten System versteht man hinsichtlich des Metacomputing, „einen Leistungsverbund aus mehreren Rechenknoten, die über ein Netzwerk miteinander verbunden sind“[166]. Ein solcher Verbund stellt eine große virtuelle Maschine dar, wobei zwischen zwei Arten, dem Parallelrechner einerseits und dem Cluster andererseits, unterschieden wird. Während ein Parallelrechner aus einer Vielzahl von Prozessoren aufgebaut ist und schon einen Leistungsverbund innerhalb der Maschine darstellt, besteht ein Cluster aus mehreren Einzelrechnern wie Workstations oder PCs, die über ein lokales Netzwerk verknüpft sind.[167]

Der Begriff „Metacomputing“ wurde erstmals für den nahtlosen Zugang von Arbeitsplatzrechnern zu Supercomputern vom Direktor des National Center für Supercomputing Anwendungen (NCSA), *Larry Smarr*, benutzt und war der eigentliche Vorgänger

---

[163] Orzech /Grid/

[164] Vgl. Orzech /Grid/

[165] Vgl. o.V. /Einleitung zum Metacomputing/

[166] o.V. /Einleitung zum Metacomputing/

[167] Vgl. o.V. /Einleitung zum Metacomputing/

des Verteilten Rechnens.[168] Im Jahr 1992 wurde dieser Begriff in einem Artikel der „Communication of ACM“ von den Autoren *Charles Catlett* und *Larry Smarr* vom National Center für Supercomputing der Universität von Illinois definiert:[169] “The computing resources transparently available to the user via this networked environment have been called a metacomputer. The metacomputer is a network of heterogeneous, computational resources linked by software in such a way that they can be used as easily as a personal computer.”[170] Dieses Netzwerk soll einen Leistungsverbund bilden, bei dem, vergleichbar mit einem Stromnetz, jeder Nutzer an jedem Ort seinen Computer über ein Kabel mit einer Steckdose verbindet, um somit weltweit auf die Computerressourcen zugreifen zu können, die zur Bewältigung seiner Problemstellung benötigt werden.[171]

Der Forschungsverbund „Verteiltes Höchstleistungsrechnen“ definiert Metacomputing als „den Zusammenschluß einzelner vernetzter Rechnersysteme zu einem virtuellen Maschinenraum, der sich nach außen als homogene Ressource darstellt“[172]. In der Fachliteratur ist jedoch bislang weitestgehend ungeklärt, ab welcher Größe ein solcher Verbund als Metacomputing bezeichnet werden kann. Hinsichtlich der oben aufgeführten Definition könnte „bereits ein Cluster vernetzter baugleicher Workstations mit einem Management-System“[173] als Metacomputer gelten.

Foster und *Kesselmann* wiederum definieren Metacomputer als „a networked virtual supercomputer, constructed dynamically from geographically distributed resources linked by high-speed networks“[174]. Im Gegensatz zu *Catlett* und *Smarr* bestimmen sie explizit, durch die Kopplung virtueller Supercomputer, welche Art von Ressourcen über Hochgeschwindigkeitsnetzwerke verknüpft werden.

Der Begriff des Metacomputing „etablierte sich während der letzten Jahre als Synonym für das Erreichen sehr hoher Rechenleistung durch das Koppeln von Hochleistungsrechnern an verteilten Orten“[175]. Diese Kopplung von Parallelrechnern, Clustern u.a. erlaubt ein verteiltes Höchstleistungsrechnen, da in diesem Leistungsverbund mehr Res-

---

[168] Vgl. o.V. /Past to Present/, Vgl. o.V. /Introduction to the Grid/

[169] Vgl. Schirmbacher /Metacomputing-Editorial/ 1

[170] Smarr, Catlett /Metacomputing/

[171] Vgl. Schirmbacher /Metacomputing-Editorial/ 1

[172] Gehring /NRW-Metacomputing/

[173] Gehring /NRW-Metacomputing/

[174] Foster, Kesselmann /Metacomputing Infrasturcture/

[175] Drum /Effiziente Methoden/ 10

sourcen zur Verfügung stehen beziehungsweise ein Lastenausgleich diese effizienter nutzen kann. Darüber hinaus stehen den einzelnen Anwendern durch den Metacomputer neue Ressourcen-Typen zur Verfügung, die lokal meist gar nicht vorhanden sind.[176] Demzufolge wird in der Fachliteratur nicht von einer Alternative zum Supercomputing sondern vielmehr von dessen logischer Fortführung gesprochen.[177]

7.2 Metacomputing im Speziellen

Die Idee des Metacomputing resultiert ursprünglich aus der Forderung der Benutzer, ihre Anwendungen auf die Architekturlinien zu verteilen, die für die jeweilige Problemstellung eine geeignete Lösungsplattform bieten.[178] Dadurch sollen die vorhandenen Ressourcen effektiver als im traditionellen Computing genutzt werden. „Metacomputer können allgemein als eine Dienste vermittelnde Schicht zwischen Ressourcen aller Art und der Anwendungen gesehen werden.“[179] In der Fachliteratur werden Metacomputer hinsichtlich dieser Vermittlerrolle auch als „Middleware“ bezeichnet,[180] wobei die Middleware lediglich den Kern oder das Tool zum Aufbau von Metacomputern darstellt. „Ähnlich der Struktur innerhalb eines Rechners, indem zwischen der Anwendung und der Hardware das Betriebssystem Dienste vermittelt, stellt eine Metacomputer-Architektur verteilte Dienste für verteilte Anwendungen zur Verfügung. ... Der Aufbau einer solchen Middleware ist bei allen Systemen in lokale und globale Komponenten unterteilt. Lokale Komponenten sind auf den beteiligten Rechnern für die Ausführung und Verwaltung der Anwendung installiert, globale Komponenten verbinden diese zu einem Gesamtsystem. Innerhalb dieses prinzipiellen Aufbaus unterscheiden sich Metacomputer sowohl von den eingesetzten Techniken als auch von der Mächtigkeit der angebotenen Dienste.“[181]

---

[176] Vgl. o.V. /Einleitung zum Metacomputing/, Vgl. Drum /Effiziente Methoden/ 22

[177] Die folgenden Ausführungen sind Vgl. Gehring /NRW-Metacomputing/ entnommen worden.

[178] Vgl. Sander /Metacomputer-Architektur/

[179] Drum /Effiziente Methoden/ 4

[180] Vgl. Drum /Effiziente Methoden/ 4

[181] Drum /Effiziente Methoden/ 4

7.2.1 Arten von Metacomputing-Systemen

„Unabhängig von der Struktur der Anwendung lassen sich Metacomputing-Systeme in Client/Server Systeme und in dezentrale, verteilte Systeme unterteilen.“[182]

Bei Client-Server-Systemen erfolgt die Verwaltung der Clients an einem zentralen Punkt, dem Server.[183] Dieser kennt die Clients und nimmt die Anfragen zur Verteilung der Ressourcen entgegen. Diese Systeme haben eine einfachere Struktur und ermöglichen ein leichteres Sicherstellen der Datenintegrität als dezentrale verteilte Systeme. Allerdings haben sie eine höhere Ausfallwahrscheinlichkeit und sind in Bezug auf die Erweiterbarkeit des Gesamtsystems von der Leistungsfähigkeit des Servers abhängig.

Dezentrale verteilte Systeme verfügen nicht über zentrale Komponenten, die Dienste koordinieren oder Informationen sammeln. Deshalb werden diese in so genannte modulare Einheiten zerlegt, um sie auf den einzelnen, beteiligten Rechnern auszuführen. Dieser Informationsaustausch lässt eine Gesamtsicht auf das System zu, die als Informationsgrundlage für die einzelnen Dienste genutzt werden kann.

7.2.2 Anforderungen an Metacomputing-Systeme

Ein großer Teil der im Bereich des Parallelen und Verteilten Rechnens zu bewältigenden Herausforderungen lassen sich auch auf Metacomputer-Umgebungen übertragen. Aus den im Folgenden dargestellten Anforderungen ist erkennbar, dass sich durch die verteilte Architektur verlagerte Probleme ergeben können, deren Lösung mit herkömmlichen Methoden allenfalls unzureichend möglich ist. Im Folgenden möchte ich näher auf die an Metacomputing-Systemen gestellten Anforderungen eingehen.[184]

7.2.2.1 Fehlertoleranz

In Standard-Rechnerarchitekturen sind nur wenige funktionell unabhängige Komponenten vorhanden, was beim Ausfall einer Komponente wie z.B. CPU, Speicher, Bussystem dazu führt, dass das gesamte System den Betrieb einstellen muss. Erst bei sehr großen, komplexen Systemen bis hin zu Supercomputern werden Fehlertoleranzmechanismen durch Techniken wie Prozessreplikation und transaktionsorientierter Verarbeitung notwendig. Dies wird vorrangig bei Systemen kommerzieller Anbieter für Datenbanken oder Netzdienste praktiziert. Im wissenschaftlich-technischen Rechnen hingegen ist dies

---

[182] Drum /Effiziente Methoden/ 20

[183] Die folgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 20,21 entnommen worden.

[184] Die Ausführungen des Kapitel 7.2.2 sind Vgl. Drum /Effiziente Methoden/ 12-16 entnommen worden.

sehr selten der Fall; hier wird die Applikation im Fehlerfall neu aufgesetzt. Aufgrund der hohen Anzahl der beteiligten Knoten und Netzkomponenten ist die Fehleranfälligkeit bei Metacomputern vergleichsweise hoch, woraus die Notwendigkeit solcher Sicherungsmaßnahmen besteht, damit lang laufende und aufwendig aufzusetzende Applikationen nicht neu gestartet werden müssen. (Vgl. Kapitel 3.1.5)

7.2.2.2 Heterogenität

Die Heterogenität zielt im Zusammenhang mit verteilten Architekturen auf die unterschiedliche Steuerung verschiedener Geräte, Rechnerhardware, Betriebssysteme und Programmiersprachen ab.

7.2.2.3 Dynamische Struktur

Im Gegensatz zu fest installierter Hardware, bei der die Konfiguration angeschlossener Geräte und der zugehörigen Software nur selten verändert wird, ist die Struktur eines Metacomputers sehr dynamisch. Aufgrund dieser Struktur können zur Laufzeit von Anwendungen weitere Rechner zum Gesamtsystem hinzugefügt oder entfernt werden. Dieses dynamische Verhalten stellt vergleichsweise hohe Anforderungen an das Ressourcenverwaltungssystem eines Metacomputers sowie an den Fehlertoleranzmechanismus und das Informationsverwaltungssystem bei Veränderungen der Architektur während der Laufzeit.

7.2.2.4 Vorhersehbare Laufzeit

Auch wenn die dynamische Struktur eines Metacomputers eine hohe Flexibilität bietet, ist es von Vorteil, Aussagen über die Laufzeit im Voraus treffen zu können. Insbesondere bei lang laufenden Anwendungen im Bereich des wissenschaftlich-technischen Rechnens sind Vorhersagen über die Laufzeit von großer Bedeutung und werden auch in Zukunft bei schwankender Knotenzahl und unterschiedlicher Rechenkapazität in Metacomputing-Umgebungen eine große Herausforderung bleiben.[185]

7.2.2.5 Rechner- und Institutionsautonomie

Aufgrund der weiten Verteilung der Rechner in Metacomputing-Landschaften können nicht alle Rechner eines Metacomputing-Systems von einem einzelnen Administrator verwaltet werden. Daraus ergeben sich Schwierigkeiten bei der Verwaltung des Ge-

---

[185] Vgl. Schumann /Automatic Performance/, Vgl. Casanova, Dongarra /A Network Enabled Server/

samtsystems bezüglich gleicher Sicherheitsbestimmungen der beteiligten Knoten sowie der einheitlichen Installation von Hard- und Software. Um diese Schwierigkeiten bestmöglich pflegen zu können ist eine direkte Zusammenarbeit der Administratoren notwendig.

7.2.2.6 Performance

Um die Eigenschaft der Transparenz verteilter Systeme (Vgl. Kapitel 3.1.6) realisieren zu können, ist ein hoher Implementierungs- und Verwaltungsaufwand bei der Verwirklichung einer transparenten Metacomputing-Umgebung notwendig. Werden alle aufgeführten Anforderungen berücksichtigt und der Metacomputer als Middleware mit zusätzlicher Redundanz zur Fehlertoleranz und den notwendigen Protokollen, um die Sicherheit und Integrität der Software zu wahren, ausgestattet, ergeben sich durch die Verwaltung dieser Dienste Leistungseinbußen. Da das eigentliche Hauptziel eines Metacompting-Systems die Leistungssteigerung einer Anwendung ist, sind Optimierungen auf der Ebene der Middleware unbedingt notwendig, um die potentielle Leistung einer solchen Umgebung bis zur Anwendung zu transportieren. Dies kann beispielsweise durch das Abschließen der Verwaltung vor dem Aufsetzen der Anwendung durchgeführt werden, um den Protokollaufwand zur Laufzeit so gering wie möglich zu halten.

7.2.2.7 Portabilität

Aufgrund der Portabilität der soeben aufgeführten Komponenten sollte sowohl die eingesetzte Middleware als auch die Anwendung portabel sein. Mittels herkömmlicher Programmiersprachen ist dies aufwendig und lediglich unter hohem Entwicklungsaufwand möglich. Mit der plattformunabhängigen Programmiersprache „Java“ und deren virtueller Maschine, der JVM, wird größtmögliche Portabilität erreicht ohne auf Basisfunktionalitäten verzichten zu müssen.

7.2.2.8 Sicherheit

Eine der bedeutendsten Herausforderungen sowohl für die Zukunft des Internets als auch im Hinblick auf den Erfolg von Metacomputing-Systemen ist bei global kommunizierenden Applikationen die Sicherheit. Aufgrund der unerlässlichen Notwendigkeit dieser möchte ich darauf in Kapitel neun gesondert eingehen.

7.2.2.9 Skalierbarkeit

Die Skalierbarkeit im Zusammenhang mit Metacomputing-Systemen ist die Fähigkeit eines Systems bei Variation eines quantitativen Parameters die wesentlichen Systemeigenschaften aufrecht zu erhalten. Demnach kann beim Hinzufügen von Komponenten wie z.B. Prozessoren die Effizienz aufrechterhalten werden. Innerhalb des Metacomputing-Systems hingegen können andere Ressourcen einen Flaschenhals bilden, der die Skalierbarkeit hinsichtlich der Leistung beschränkt. Dies kommt insbesondere bei Client/Server-Architekturen in dem Moment vor, wenn die Anzahl der Clients die Netzkapazität übersteigt. Deshalb werden Metacomputing-Systeme verteilt administriert und häufig ohne zentrale Komponente realisiert. Dadurch erhöht sich die Komplexität des Gesamtsystems, die mit zusätzlichem Aufwand, in Form teurer Kommunikation, ausgeglichen werden muss.

7.2.2.10 Lastverteilung

Im Vergleich zu herkömmlichen verteilten Architekturen werden in Metacomputing-Systemen Mechanismen zur Lastverteilung dann eingesetzt, wenn große Lastungleichheiten vermieden werden sollen, da der langsamste Prozess die Gesamtlaufzeit einer Metacomputing-Anwendung bestimmt. Das Gebiet der Lastverteilung unterliegt aufgrund der vergleichsweise hohen Komplexität intensiven Forschungstätigkeiten, da deren Ziele[186] sehr vielfältig sind. Im Hinblick auf Metacomputing ist das Hauptziel der Lastverteilung, den schnellsten beteiligten Knoten herauszufinden und mit dem rechenintensivsten Prozess zu versorgen.

7.2.3 Dienste von Metacomputing-Systemen

Die von Metacomputing-Systemen implementierten Dienste sind größtenteils bereits in anderen Umgebungen des Verteilten oder Parallelen Rechnens zu finden.[187] Jedoch stellen nahezu alle Systeme der Anwendung oder dem System selbst Mechanismen zur verteilten Ausführung zur Verfügung, deren Funktionen in der Regel nicht orthogonal sind,

---

[186] Die Ziele der Lastverteilung variieren stark und können abhängig vom Fokus der Anwendung ein hoher Durchsatz, Fehlertoleranz oder Laufzeitminimierung sein.

[187] Vgl. Drum /Effiziente Methoden/ 26

so dass sich Überschneidungen ergeben.[188] Auf die grundlegenden Dienste einer Metacomputing-Umgebung möchte ich im Folgenden näher eingehen.[189]

Die „Prozessverteilung“ stellt einen sehr wichtigen Dienst eines Metacomputers dar, indem sie durch die Verteilung der Last auf mehrere Knoten eine der Grundfunktionen des Metacomputers übernimmt. Diese Verteilung kann implizit durch das Erzeugen von Prozessen bzw. Funktionen oder durch einen expliziten Aufruf zur Verteilung in der Anwendung erfolgen. Ähnlich wie bei der Kommunikation bietet der implizite Ansatz eine höhere Abstraktionsebene an, durch den der Entwicklungsaufwand reduziert werden kann.

Die meisten Metacomputer-Architekturen bieten „Kommunikationsdienste“ an, welche analog zur Prozessverteilung, implizit oder explizit ausgeführt werden können. Dabei werden im wissenschaftlich-technischen Rechnen explizite Kommunikationsaufrufe wegen der Geschwindigkeitsvorteile bevorzugt, jedoch gibt es auch die Bestrebungen objektorientierte Ansätze ohne Leistungsverlust einzusetzen. Bei impliziten Kommunikationsmodellen wird im Allgemeinen kein Dienst an sich angeboten, vielmehr wird die Kommunikation innerhalb von Methodenzugriffen auf entfernte Objekte realisiert.

Die Aufgaben der „Lastverteilung“ erfordern innerhalb von Metacomputing-Umgebungen eine gleichmäßige Verteilung von Prozessen auf Knoten[190], wobei stets die Forderung zugrunde liegt, dass performante Knoten möglichst ausgelastet sein sollen, wenn ein Auftrag zur Berechnung bereit steht. In Metacomputing-Systemen mit sehr hoher Knotenanzahl sollte die Lastverteilung nahezu unproblematisch sein, da pro Knoten jeweils ein Prozess zugeordnet werden kann. Aufgrund der unterschiedlichen Leistung der einzelnen Knoten können länger laufende Teilaufgaben an Prozessoren mit hoher Leistung gebunden werden.

Die „Prozessmigration“ bietet die Möglichkeit, Prozesse auf andere Knoten zu übertragen. Die Motive dafür können lokal nicht verfügbare Ressourcen, insbesondere eine Lastungleichheit, sein. Die Sicherungspunkterzeugung wird in diesem Zusammenhang dafür genutzt, an ausfallende Ressourcen vergebene Teilaufgaben nicht erneut vollständig berechnen zu müssen.

---

[188] Vgl. Drum /Effiziente Methoden/ 26, 27

[189] Die folgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 27, 28 entnommen worden.

[190] Als ein Knoten wird im Zusammenhang mit Metacomputing-Umgebungen ein Rechner bezeichnet.

Da komplexe Metacomputer-Systeme über keine zentrale Komponente der Informationsgewinnung verfügen, unterstützt der Dienst der „Informationsgewinnung“ diese Systeme bei der Erkennung neuer Komponenten zur Laufzeit und bei der Vermittlung der Informationen zu bekannten Komponenten. Ein Grund für das Fehlen einer solchen Komponente könnte die begrenzte Skalierbarkeit solch komplexer Systeme sein.

Die Dienste zur „Sicherheit“ umfassen die Identifikation, Authentifizierung und sichere Kommunikation z.B. durch Verschlüsselung. Die Benutzung von entfernten Ressourcen erfordert folgerichtig einen erhöhten Protokollaufwand beim Zugriff auf fremde Ressourcen und zusätzliche Kommunikation, die gewöhnlich den entscheidenden Faktor für den Geschwindigkeitsvorteil darstellt. Insbesondere bei Anwendungen des wissenschaftlich-technischen Rechnens, die auf die Optimierung der Rechenleistung abzielen, wirkt die Verschlüsselung der zu übertragenden Datenpakete leistungsmindernd. Allerdings sollten gerade bei Projekten mit einer hohen Privatbeteiligung wie z.B. SETI@home die Vor- und Nachteile zwischen gewonnener Leistung und notwendiger Sicherheit abgewogen werden.

„Visualisierungsdienste“ bieten die Möglichkeit, entfernte Daten zu aktuellen Berechnungen zur Laufzeit der Anwendung abzugreifen, diese zu einer entsprechend ausgestatteten lokalen Komponente zu senden und anschließend darzustellen. Dies können neben Berechnungsergebnissen der Anwendung auch der Programmablauf oder der Systemzustand des Metacomputers sein. Um die vergleichsweise hohen Anforderungen an die Bandbreite bezüglich der visuellen Informationen zu erfüllen, ist es notwendig, zur Platzierung der beteiligten Prozesse, Informationen zur Netztopologie einzubeziehen.

Integrierte Dienste zur „Beobachtung“ der Anwendung und des Metacomputer-Systems selbst ermöglichen in komplexen verteilten Umgebungen Analysen zum Laufzeitverhalten, zu Fehlern und Leistungsengpässen. Aufgrund der zu erwartenden Entwicklungsprobleme hinsichtlich der Fehleranfälligkeit bei Metacomputern sollten die Anforderungen an Werkzeuge zur Beobachtung und Manipulation der Anwendung verstärkt werden.

7.3 Einordnung des Metacomputing

In Abb. 7-1 wird das Konzept des Metacomputing vor dem Hintergrund des Parallelen und Verteilten Rechnens grafisch dargestellt. Wie bereits in Kapitel vier ausgeführt, haben beide Konzepte neben ihren Unterschieden auch signifikante Gemeinsamkeiten.[191]

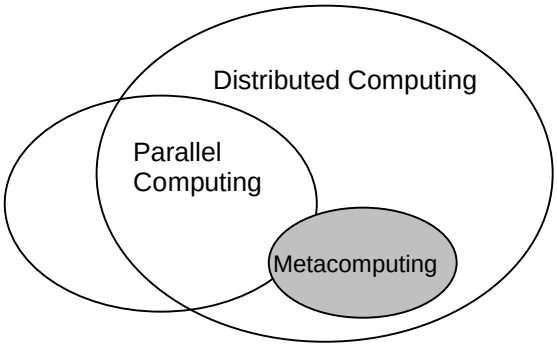

Abb. 7-1: Einordnung des Metacomputing[192]

Auch Metacomputer haben Gemeinsamkeiten und Unterschiede mit parallelen und verteilten Systemen.[193] Ähnlich wie verteilte Systeme muss ein Metacomputer, auch als Verknüpfung mehrerer Supercomputer bezeichnet, in der Lage sein, an unsichere Netzwerke gebundene Ressourcen mit verschiedenen Fähigkeiten zu integrieren. Demnach können aufgrund der erforderlichen hohen Performance Programmiermodelle und -schnittstellen benötigt werden, die sich von der Architektur verteilter Systeme unterscheiden. Wie in parallelen Systemen sollte in Metacomputern auf eine effiziente Kommunikation geachtet werden, um die erforderliche Performance erreichen zu können.

Die effektive Leistung eines Metacomputers ist sehr von der Art der Anwendung abhängig.[194] Wie schon bei anderen Hochleistungsarchitekturen bekannt, bevorzugen Multirechner-, Mehrprozessorsysteme oder Cluster bestimmte Applikationen beziehungsweise deren Berechnungen oder Kommunikationsmuster.

In Metacomputer-Architekturen hingegen ist die Ausführungsplattform im Allgemeinen nicht bekannt. Darüber hinaus „limitieren die beschränkte Bandbreite und die hohen La-

---

[191] Vgl. Leopold /Computing/ 4

[192] Vgl. Leopold /Computing/ 5

[193] Die folgenden Ausführungen wurden Vgl. Foster, Kesselmann /Metacomputing Infrasturcture/ entnommen.

[194] Vgl. Drum /Effiziente Methoden/ 2

tenzzeiten des Verbindungsnetzwerkes die Möglichkeiten der Parallelisierung“[195]. Eine sich aus diesen Eigenschaften ergebende logische Folge ist die effiziente Ausführung lediglich für Anwendungen mit einer hohen Parallelität. „Die Kommunikation in parallelen oder verteilten Anwendungen ist eine der wichtigsten Faktoren für die Effizienz einer Parallelisierung.“[196] Aufgrund der außerordentlich hohen Anzahl an Ressourcen hat die effiziente Auslastung bei Metacomputern nicht den Stellenwert anderer dedizierter Hochleistungsrechner.[197] So stehen einerseits eine möglichst hohe und gleichmäßige Auslastung der Prozessoren und andererseits eine möglichst kurze Rechenzeit und die Möglichkeit besonders umfangreiche Aufgabenstellungen zu berechnen, im Vordergrund. Zudem tendieren Metacomputer-Architekturen grundsätzlich zu einer Art Gruppenbildung; einer Anzahl von Rechnern mit einer zueinander guten Netzanbindung, die zu einer oder mehreren Gruppen eine weniger gut ausgebaute Verbindung unterhält. Dafür geeignete Anwendungen sind beispielsweise so genannte „coupled fields“-Algorithmen, die von der Metacomputer-Umgebung erkannt werden und die Prozessverteilung dementsprechend vornehmen.

Unter Berücksichtigung der aufgeführten Argumente können parallele und verteilte Systeme als Basis von Metacomputern verstanden werden, erfordern jedoch zusätzliche Weiterentwicklungen der Mechanismen, Techniken und Werkzeuge.[198] Das „Globus“-Projekt beschäftigt sich innerhalb dieser Weiterentwicklung mit dem langfristigen Ziel sowohl die Probleme der Konfiguration als auch die Optimierung der Performance auf Metacomputer-Umgebungen zu übertragen.

8 Analyse von Metacomputing-Systemen

In diesem Kapitel möchte ich eine Übersicht über gegenwärtig existierende Metacomputing-Systeme geben. Dabei möchte ich zunächst den Versuch einer Klassifikation dieser Systeme - ausgehend von einfachen Umgebungen bis hin zu komplexen und aufwendigen Systemen - vornehmen. Im Anschluss werde ich sowohl auf Metacomputing-Systeme zum wissenschaftlich-technischen Rechnen, auch als einfache Umge-bungen bezeichnet, als auch auf das ausgereifte Metacomputing-System Legion detaillierter eingehen.

---

[195] Drum /Effiziente Methoden/ 2

[196] Drum /Effiziente Methoden/ 2

[197] Die folgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 3 entnommen worden.

[198] Vgl. Foster, Kesselmann /Metacomputing Infrasturcture/

8.1 Klassifikation von Metacomputing-Systemen

Die Anzahl verteilter Systeme ist in den letzten Jahren im Zuge der Ausbreitung des Internet und der zunehmenden Vernetzung von Computersystemen stark angestiegen.[199] Diese Entwicklung gilt sowohl für Systeme mit lokal verteilten Ressourcen als auch für global verteilte Systeme. Im Zusammenhang mit der Thematik des Metacomputing können anhand der Komplexität verschiedene Stufen von Metacomputing-Systemen unterschieden werden. Diese Klassifikation reicht von „Systemen zur Nutzung trivialer Parallelität“ über „Java basierte Client-Server-Umgebungen“, „Atlas“ und „Legion“ bis hin zum „Globus-System“.

Dabei umfassen Systeme zur Nutzung trivialer Parallelität, in der Literatur auch als Metacomputing-Systeme zum wissenschaftlich-technischen Rechen bezeichnet, Projekte wie beispielsweise „SETI@home“ und „Distributed.net“.

Das Ziel Java basierter Client-Server-Umgebungen, die Systeme wie „Javelin“ und „JET“ enthalten, war es, „Metacomputing auf vielen heterogenen Maschinen zu realisieren, indem sie sich auf Java Applets und das HTTP-Protokoll abstützen“[200].

Das Metacomputer-System Atlas der Universitäten von Kalifornien, Berkeley und Texas, Austin dient der Ausnutzung weltweiter Rechnerressourcen in und zwischen den Instituten. Im Gegensatz zu den vorherigen Projekten beruht Atlas nicht auf der Ausnutzung trivialer Parallelität, denn die Laufzeitumgebung ist als Middleware zwischen Applikation und Betriebssystem angelegt. Darüber hinaus dienen „Java“ und „Cilk“[201] als Grundlage.

Legion ist ein Projekt des „Departments of Computer Science“ der Universität Virginia und mehreren strategischen Partnern.[202] Es stellt derzeit das wohl vollkommenste verfügbare Metacomputing-System dar und wurde bereits auf mehreren Testumgebungen installiert.[203]

Das Globus-System ist ein Projekt der Universität Südkaliforniens von *Foster* und *Kesselmann*.[204] Dieses Projekt „soll einen Schritt in die Richtung eines Computational

---

[199] Die nachfolgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 39 ff. entnommen worden.

[200] Drum /Effiziente Methoden/ 43

[201] Cilk ist eine auf C basierende Programmiersprache, welche die parallele Ausführung von Threads ermöglicht. Vgl. Drum /Effiziente Methoden/ 47

[202] Vgl. o.V. /Legion/

[203] Vgl. Drum /Effiziente Methoden/ 48

[204] Vgl. o.V. /Globus/

Grid[205] realisieren, indem die hierfür notwendigen Dienste entwickelt“[206] werden und „eine prototypische Testumgebung auf diesen Diensten aufgebaut“[207] wird. Schließlich sollen Anwendungen diese Umgebung nutzen.

8.2 Metacomputing-Systeme zum verteilten Rechnen

So genannte PC-Cluster erfreuen sich zunehmend hoher Beliebtheit und werden gegenwärtig für rechenintensive und parallele Anwendungen eingesetzt. Dabei kann die aus der Verknüpfung moderner Arbeitsplatzrechner gewonnene Rechenleistung diejenige von herkömmlichen, „echten“ Parallelrechnern zum einen nominell erreichen und darüber hinaus ein deutlich besseres Preis-Leistungs-Verhältnis gewährleisten.[208]

8.2.1 SETI@home

„SETI“ - „Search for ExtraTerrestrial Intelligence“ ist ein Projekt der Universität von Kalifornien, Berkeley, welches sich auf wissenschaftlicher Basis mit der Suche nach fremden Lebensformen im Universum beziehungsweise nach außerirdischer Intelligenz beschäftigt.[209] In den letzten Jahren ist eine Vielzahl verschiedener wissenschaftlicher Teams entstanden, welche auf unterschiedlichste Art und Weise nach außerirdischer Intelligenz suchen. Einige dieser Teams suchen nach Signalen in den pulsierenden Lichtstrahlen, die von Sternen abgestrahlt werden, die Mehrzahl jedoch sucht in Milliarden von Radiofrequenzen, die das Universum durchfluten, ob es eine Zivilisation gibt, die möglicherweise Radiosignale überträgt.[210] Dabei gilt es „Radiosignale verschiedener Frequenzen zu allen Zeitpunkten von diversen Punkten aus dem Weltall aufzuzeichnen und auszuwerten“[211]. Auf dieser Grundlage und der Tatsache, dass Großcomputer aufgrund der großen Datenmenge nicht in der Lage sind, die Daten detailliert auf schwächere Signale beziehungsweise eine große Bandbreite verschiedener Signal-Typen zu untersuchen, ist im Mai 1999 das Projekt SETI@home entstanden. Dieses soll potentiel-

---

[205] „Ein Computational Grid ist eine Infrastruktur aus Hardware und Software, die eine Steigerung der Rechenleistung ermöglicht, indem sie konsistenten, zuverlässigen, allgegenwärtigen und kostengünstigen Zugang zu Hochleistungsrechenanlagen bietet.“ Drum /Effiziente Methoden/ 22, Vgl. Kapitel 6

[206] Drum /Effiziente Methoden/ 53

[207] Drum /Effiziente Methoden/ 53

[208] Vgl. Maehle /Hochleistungs-PC-Cluster/

[209] Vgl. o.V. /Was ist SETI@home?/

[210] Die folgenden Ausführungen sind Vgl. o.V. /SETI/ entnommen worden.

[211] o.V. /Was ist SETI@home?/

len Teilnehmern die Möglichkeit bieten „@home“ - von Zuhause aus - die Rechenleistung ihrer Computer zu nutzen, um die Analyse der aufgenommenen Radiowellen aus dem Weltall zu unterstützen.

8.2.1.1 Architektur und Funktionsweise

SETI@home benutzt bei der Suche nach außerterrestrischen Radiosignalen das wohl größte Teleskop der Welt, das so genannte „Arecibo-Teleskop“ in Puerto Rico, Südamerika (nahe der Stadt Arecibo), um den Himmel für Funksignale abzulichten.[212] Dieses Teleskop „ist ein nicht schwenkbarer sphärischer Reflektor mit einem Durchmesser von über 305 Metern“[213], der „in ein natürliches, schüsselförmiges Tal eingelassen ist“[214].

Dieses Tal reflektiert und bündelt die schwachen Himmelssignale in den Empfangsantennen, die sich circa 137 Meter über dem Boden befinden.[215] Aufgrund der Tatsache, dass der Reflektor nicht geschwenkt werden kann, sind die Empfangsantennen an einer bogenförmigen Schiene befestigt, um so alle Objekte in einem Umkreis von 20 Grad ab dem Zenit erfassen zu können. Dieser bogenförmige Arm ist an einer kreisförmigen Konstruktion befestigt, die mit langen Drahtseilen von drei Trägergestellen gehalten wird. Daher können die Antennen aufgrund der Erdrotation ein Objekt verfolgen, während es den Himmel durchquert. Diese Bewegungen ermöglichen es dem Teleskop, einen relativ großen Bereich des Himmels zu durchsuchen.

Die vom Radioteleskop Arecibo aufgenommenen Daten werden dafür in Pakete aufgeteilt, um sie den potentiellen Projektteilnehmern per Download zur Verfügung stellen zu können.[216] Da hierbei enorme Datenmengen anfallen, die bis auf die grundlegende Partitionierung des Suchraumes unabhängig voneinander berechnet werden können, bietet sich eine parallele, verteilte Abarbeitung an. (Vgl. Kapitel 4) Um die in diesem Zusammenhang entstandenen Herausforderungen bewältigen zu können, ist eine Infrastruktur geschaffen worden, die es ermöglicht, diese Daten auf weltweit verteilten Rechnern zu analysieren.[217] In Abb. 8-1 sind der Datenfluss der vom Arecibo-

---

[212] Vgl. o.V. /SETI/

[213] Hipschman /Radioteleskop/

[214] o.V. /Arecibo/

[215] Die folgenden Ausführungen sind Vgl. o.V. /Radioteleskop/, Vgl. o.V. /Arecibo/ entnommen worden.

[216] Vgl. o.V. /SETI@Home I/

[217] Vgl. Drum /Effiziente Methoden/ 41

Radioteleskop eingefangenen Daten, die Zerlegung dieser in so genannte „work units“ und der anschließende Versand zum bearbeitenden Client grafisch dargestellt.

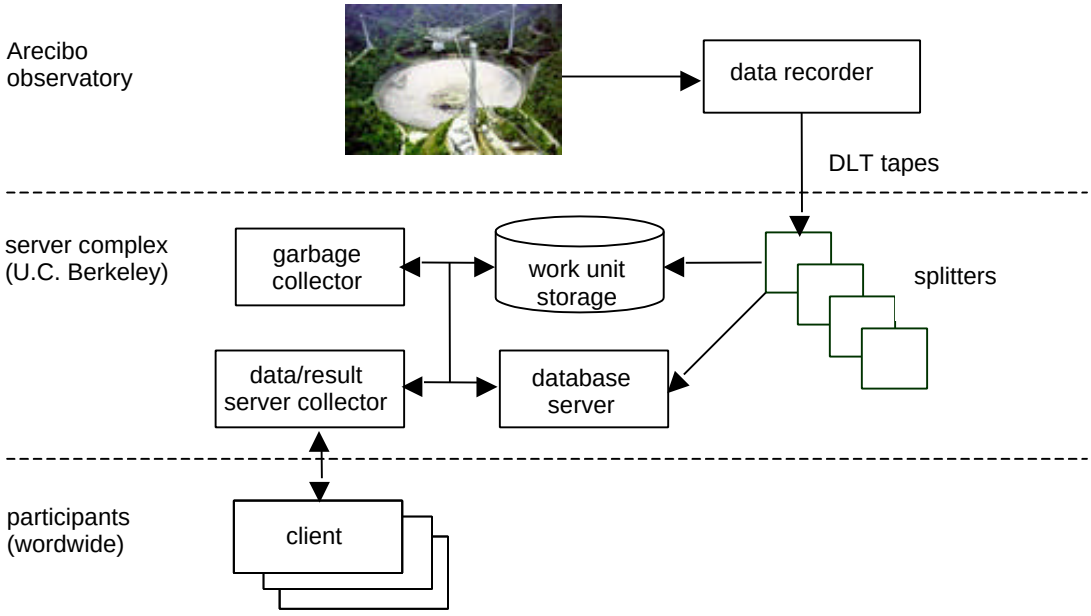

Abb. 8-1: Der Datenfluss der empfangenen Daten[218]

Da die Stadt Arecibo nicht über eine Breitband-Internetverbindung verfügt, müssten die Datenbänder über ein 56K-Modem[219] nach Berkeley gesendet werden. Aufgrund dieser Tatsache wurde entschieden, die eingefangenen Daten mit hoher Dichte aufzuzeichnen, um die Datenmengen besser bewältigen zu können. Die dafür eingesetzten Datenträger sind so genannte „DLT-Cartridges“ mit einem Speichervolumen von 35 GByte, was dem Aufnahmevolumen eines ganzen Tages entspricht.[220] In der verhältnismäßig einfachen Handhabung der DLT-Tapes liegt auch ein sehr wesentliches Motiv für den Einsatz zentralisierter Serverfunktionalitäten für das Projekt SETI@home.[221] Dabei wurde für die Aufzeichnung eine Rate von 5 MBit pro Sekunde gewählt, die einerseits niedrig genug ist um die Aufzeichnungszeit überschaubar zu halten und die Daten über die laborinterne 100 MBit-Internetverbindung verteilen zu können und andererseits hoch genug, um mit dieser Datenbasis eine signifikante Analyse durchführen zu können.

---

[218] Vgl. Anderson et al. /An Experiment/

[219] Die Kommunikation über ein 56K-Modem (KBit pro Sekunde) wird aufgrund seiner vergleichsweise niedrigen Datenübertragungsrate gegenüber Breitbandnetzen auch als Schneckenpost-„snail-mail“ bezeichnet.

[220] Vgl. Hipschman /Wie funktioniert SETI@home?/

[221] Die folgenden Ausführungen sind Vgl. Anderson et al. /An Experiment/ entnommen worden.

Nach erfolgter Aufzeichnung wird das aufgenommene Signal, das eine Bandbreite von 2,5 MHz umfasst, mit einem Programm namens „Splitter“ in 256 10-KHz Segmente zerlegt.[222]

Diese Zerlegung wird vorgenommen, um die umfangreichen Daten bearbeiten zu können. Um Signale bis zu 10 KHz registrieren zu können, müssen diese mit einer Rate von 20.000 KBit pro Sekunde abgetastet werden. Von diesen 10KHz (20KBit/s) werden circa 107 Sekunden, multipliziert mit 20.000 Bit (20KBit), in eine so genannte work unit (Arbeitseinheit) gepackt, welche zunächst eine Größe von rund 250KB erreicht. Dieses Datenpaket, mit zusätzlichen Verwaltungsdaten versehen, erreicht eine Größe von circa 340 KB und kann auf Anfrage der teilnehmenden Knoten an diese versendet werden.

Zur Ablage und Verwaltung der verschiedenen Informationen der Datenbänder, Arbeitseinheiten, Ergebnisse, Teilnehmer u.a. wird eine relationale Datenbank verwendet.[223] Der so genannte „data/result-server“ aus Abb. 8-1 ist schließlich für die Verteilung der work units an die potentiellen Teilnehmer zuständig. Der „garbage collector“ arbeitet als eine Art „Müllsammler“ mit der Aufgabe, verbrauchte work units zu löschen. Folglich stellt dieses Modul ein gewisses Gleichgewicht zu den Komponenten Server und Splitter dar, weil genauso schnell work units generiert wie gelöscht werden und ebenso schnell wieder gelöscht werden wie Ergebnisse ankommen.

Um die angeforderten Datenpakete analysieren zu können, müssen sich die Teilnehmer ein Programm herunterladen[224], welches inzwischen für nahezu jedes Betriebssystem und jede Architektur verfügbar ist.

Das SETI@home Client-Programm ist in der Programmiersprache C++ geschrieben und beinhaltet ein plattformunabhängiges System für das Verteilte Rechnen, Komponenten mit plattformspezifischen Implementierungen sowie spezifischen Code zur Analyse und grafischen Darstellung der Daten.[225] Der Client kann als ein Hintergrundprozess, als eine GUI[226]-Anwendung oder als Bildschirmschoner ausgeführt werden. Um diese verschiedenen Verfahren auf einer flexiblen Plattform unterstützen zu können,

---

[222] Die nachfolgenden Ausführungen sind Vgl. Hipschman /Wie funktioniert SETI@home?/ entnommen worden.

[223] Die folgenden Ausführungen sind Vgl. Anderson et al. /An Experiment/, o.V. /Technische Neuigkeiten/ entnommen worden.

[224] Diese Clients können auf der Website http://www.setiathome.ssl.berkeley.edu/download.html herunter geladen werden.

[225] Die folgenden Ausführungen sind Vgl. Anderson et al. /An Experiment/ entnommen worden.

[226] Die Abkürzung GUI steht für Graphical User Interface.

wurde eine Architektur entwickelt, welche einen Prozess für die Kommunikation und die Datenverarbeitung, einen weiteren für die Handhabung der GUI-Kommunikation und einen dritten für die Darstellung der Grafiken unterstützt.

Auf Clientseite zerlegt dieses Client-Programm die in der vom „data/result-server“ empfangenen work unit enthaltenen Signale mittels Verwendung einer „Fast Fourier Transformation“.[227] Dies ist notwendig, um regelmäßige oder pulsierende Signale innerhalb eines 12 Sekunden-Intervalls, die ein Punkt benötigt, um den Messfokus des Teleskops zu durchqueren, zu finden. Die Rechenzeit für diesen Prozess beträgt für eine Pentium III-CPU mit einer Taktfrequenz von 600 MHz circa 10 Stunden, da der Rechenaufwand, um schwache Signale zu lokalisieren, sehr hoch ist. Die Analyse der in den Arbeitseinheiten enthaltenen Daten kann im Hintergrund stattfinden, wenn der PC des Teilnehmers nicht vollständig ausgelastet ist.

Ein grundlegendes Merkmal des Metacomputing ist es, nur dann eine Internet-Verbindung aufbauen zu müssen, wenn Daten transferiert werden sollen.[228] Demzufolge ist eine permanente Internetverbindung nicht notwendig. Sobald die Analyse einer work unit abgeschlossen ist, wird der Teilnehmer gefragt, ob er die Daten zurücksenden und eine weitere Arbeitseinheit anfordern möchte. Diese Funktionalität kann manuell oder automatisch erfolgen und erlaubt es, stets zu überprüfen, wann der Computer eine Verbindung ins Internet aufbaut. Mit Hilfe einer Datenbank wird vom Institut in Berkeley überwacht, wo sich die jeweiligen work units gerade befinden, um sicher zu stellen, dass keine Informationen verloren gehen.

Der „data/result-server“ nimmt die vom Client bearbeiteten work units entgegen und legt sie in einer Datenbank ab.[229] Ein Programm liest diese Dateien ein, erstellt die Ergebnisse und leitet die Aufzeichnungen in eine Datenbank. Um diesen Prozess zu optimieren, werden einzelne Kopien dieser Dateien gleichzeitig ausgewertet. Für jedes Ergebnis werden vom Server Informationen über die entsprechenden Teilnehmer, deren work units, deren Rechenzeit und andere gespeichert.

Ein weiteres Programm liest diese so genannten „log-files“ ein, aktualisiert alle relevanten Datenbankeintragungen wie beispielsweise Nutzer, Team, CPU-Typ in einem Pufferspeicher und schickt sie zur Online-Datenbank. Darüber hinaus prüft ein Programm

---

[227] Die nachfolgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 41 entnommen worden.

[228] Die nachfolgenden Ausführungen sind Vgl. Hipschman /Wie funktioniert SETI@home?/ entnommen worden.

[229] Die folgenden Ausführungen sind Vgl. Anderson et al. /An Experiment/ entnommen worden.

(„redundancy elimination“) die Daten auf redundante Signale hinsichtlich der Signalanzahl und deren Parameter, filtert diese heraus und nutzt eine ähnliche Methode, um ein anerkanntes Ergebnis für diese work unit zu wählen. Diese autorisierten Ergebnisse werden auf einer separaten Datenbank („science database“) abgelegt.
Die Abschlussphase - das „back-end processing“ - besteht aus verschiedenen Schritten. Um das System verifizieren zu können, werden die Signale getestet, die am Radioteleskop aufgenommen wurden, mit dem Ziel künstlich geschaffene Signale zu erkennen und zu eliminieren.
In Abb. 8-2 ist der soeben beschriebene Empfang der Daten aus Sicht des SETI@home-Servers, die Analyse dieser Daten und die anschließende Publikation von Informationen und Ergebnissen anhand einer Grafik abgebildet.

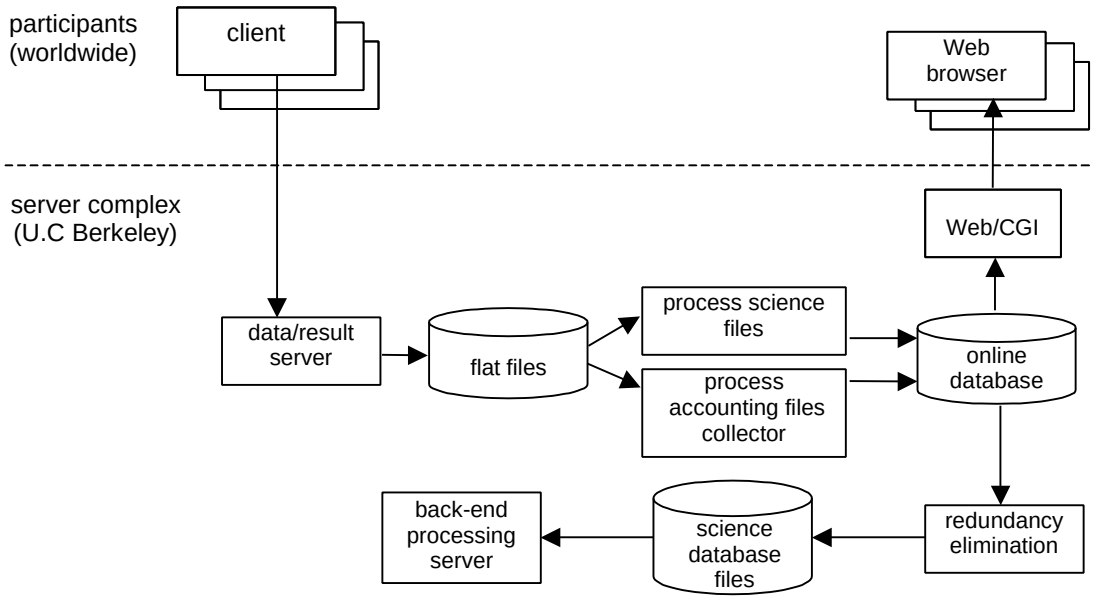

Abb. 8-2: Der Empfang und die Analyse der Ergebnisse[230]

Das Projekt SETI@home sucht nach zu verschiedenen Zeitpunkten ermittelten Signalen mit ähnlichen Frequenzen und Himmelskoordinaten. Sich wiederholende und einmalige Signale werden weiter untersucht und anhand eines akzeptierten Protokolls mit anderen SETI-Radioprojekten verglichen. Nach erfolgreicher Bearbeitung wird die entsprechende Arbeitseinheit als abgeschlossen vermerkt und dem Teilnehmer, bei Bedarf, eine weitere work unit übermittelt.

[230] Anderson et al. /An Experiment/

Die SETI@home-Website bietet allgemeine Informationen über das Projekt selbst sowie über beteiligte Projektteams, die Möglichkeit, dass zur Teilnahme benötigte Client-Programm herunter zu laden und wissenschaftlich-technische Neuigkeiten. Dabei sollen aktuelle wie auch potentielle Teilnehmer dieses Projektes hinreichend mit Informationen versorgt werden, die benutzerfreundlich im Webbrowser angezeigt werden können. Darüber hinaus werden gute Kandidaten bei der Suche nach außerirdischer Intelligenz auf dieser Website publiziert.

8.2.1.2 Nutzungsstatus und Ergebnisse

Der offizielle Start des Projektes SETI@home war am 17. Mai 1999 und kann mit derzeit circa 3,5 Millionen Teilnehmern und 1,3 Millionen Jahren CPU-Rechenzeit seit seinem Bestehen als das größte Projektrechennetzwerk der Welt bezeichnet werden.[231] Dies entspricht einer Rechenleistung von ungefähr 30 TeraFLOP pro Sekunde und einer Anzahl von etwa 500.000 Rechnern der Athlon 1GHz-Klasse, die gegenwärtig kontinuierlich für das SETI@home Projekt im Einsatz sind.[232]

Ein Problem bei der Suche nach intelligentem Leben in den Weiten des Weltraumes stellt die zunehmende „Verschmutzung des Äthers durch terrestrischen Unterhaltungs- und Kommunikationsfunk“[233] dar. So mussten nicht selten viel versprechende Signale als „irdische Radio Pollution“ verworfen werden.[234] Jedes von den Teilnehmern zurückgesendete Resultat enthält Informationen über alle signifikanten Spitzen der Gaußschen Signale, die während der Analyse ermittelt wurden.[235] Der Ausdruck signifikant wird benutzt, um Signale zu beschreiben, die hinsichtlich ihrer Geräusche unterscheidbar sind. Auch wenn bislang keine Erfolge hinsichtlich des Auffindens außerirdischer Intelligenz verbucht werden konnten, sind bereits eine Vielzahl verschiedener Signale gefunden worden, die nach sorgfältiger Analyse auf ihre Echtheit hinsichtlich ihres extraterrestrischen Ursprungs überprüft werden müssen.

Darüber hinaus können die zunehmend schnellere Verarbeitung der Arbeitseinheiten und die wachsende CPU-Rechenzeit als Fortschritte bezeichnet werden.

---

[231] Vgl. o.V. /SETI/

[232] Vgl. o.V. /Allgemeines/

[233] Gleich /Gleichschaltung/ 202

[234] Vgl. Gleich /Gleichschaltung/ 202

[235] Vgl. o.V. /Erfolge/

Bei SETI@home wird mit Hilfe des Radioteleskops in Arecibo circa ein Drittel „des gesamten Himmels beobachtet, allerdings ausschließlich über der Nordhalbkugel“[236]. Ab Ende 2003 ist das Projekt „SETI@home II“ geplant, welches mit einem weiteren Radiowellenrecorder die gesamte südliche Hemisphäre überwachen soll. Der Standort des damit größten Radioteleskops auf der Südhalbkugel wird das „Parkes Observatorium“ in Australien sein.[237] Eigens für dieses Projekt wird ein neuer Radiorecorder installiert, der in der Lage, ist Daten des Himmels von dreizehn verschiedenen Orten aus gleichzeitig abzulichten.[238] Mit diesem Projekt sollen die potentiellen Teilnehmer gleichzeitig eine neue Client-Software erhalten. Die neue Version der Software bringt dafür zusätzliche Fähigkeiten: Einerseits sollen mit Hilfe der Infrastruktur „BOINC - Berkeley Open Infrastructure for Network Computing“ ohne Wissen und Mitwirkung der Anwender „neue Versionen zu den Komponenten der Client-Software herunter geladen werden können, andererseits resultiert aus der neuen Client-Architektur auch die Fähigkeit, lokale Rechenpower nach Maßgabe des Spenders simultan auf mehrere Aufgaben zu verteilen. Damit lassen sich Ausfallzeiten, etwa wie während der Arecibo-Abschaltung, zu Gunsten anderer Datenlieferungen nutzen und es ergibt sich eine Demokratisierung der Forschungsanstrengungen, weil die Teilnehmer selbst Einfluss darauf nehmen können, welche Projekte die meiste Rechenzeit erhalten.“[239]

Grundsätzlich bietet das Projekt SETI@home Erfolge in zweifacher Hinsicht: Zum einen hilft es dabei, der Öffentlichkeit den Sinn dieses Projektes zu vermitteln und zu zeigen, auf welche Weise nach außerirdischer Intelligenz gesucht werden kann, z.B. mit Radioteleskopen.[240] Zum anderen erweist es sich als nützlich bei der Auswertung und Analyse der immensen Datenmengen. Um die bei der Bearbeitung der über 100 Millionen Frequenz-Kanäle entstehende Datenflut analysieren zu können, werden Rechenkapazitäten benötigt, die sich ein überwiegend privatwirtschaftlich finanziertes Projekt wie SETI@home nicht leisten könnte. Dieser finanzielle Nachteil wurde durch die zunehmende Beteiligung der Bevölkerung nahezu ausgeglichen.

---

[236] o.V /Rechenkraft/

[237] Vgl. o.V. /Rechenkraft/

[238] Vgl. o.V. /SETI@home II/

[239] o.V. /News bei SETI@home/

[240] Vgl. Drake /SETI@home-Erfolge/

8.2.2 Distributed.net

Das Projekt „Distributed.net“ ist eine Vereinigung zur Forschung auf dem Gebiet des verteilten Rechnens und wurde im Jahr 1997 gegründet.[241] Es kann als der „Urvater“ des Distributed Computing bezeichnet werden.[242] Die Gemeinschaft von Distributed.net hat seitdem die ersten großen internetbasierten „Distributed Computing-Projekte“ durchgeführt, bei denen es sich anfangs ausschließlich um Verschlüsselungswettbewerbe handelte. Der eingesetzte Client war dabei von Beginn an sehr ausgereift, stabil und auf verschiedenen Plattformen einsetzbar.

Das Hauptziel dieses Projektes ist es, ähnlich wie bei SETI@home, durch die Nutzung der Rechenressourcen von Privatanwendern, die nur begrenzte Zeit mit dem Internet verbunden sind, die Möglichkeiten des verteilten Rechnens aufzuzeigen. Während bei SETI@home lediglich eine Aufgabe, nämlich die Suche nach außerirdischer Intelligenz mittels Aufzeichnung von Radiosignalen, bearbeitet wird, bietet Distributed.net die Möglichkeit, unterschiedliche Problemstellungen, so genannte generische Aufgaben, in verschiedenen Teilprojekten durchzuführen. Aufgrund der Tatsache, dass bei SETI@home nur eine Aufgabe berechnet wird, bei Distributed.net jedoch verschiedene Aufgabentypen bearbeitet werden können, wird letzteres in der Praxis auch als das „echtere“ Metacomputing bezeichnet.

Das Team von Distributed.net hat sich zum Ziel gesetzt, die Entwicklung und die Förderung des verteilten Rechnens zu unterstützen. Dazu bieten sie die Möglichkeit mit jeglicher Hardware und nahezu jedem Betriebssystem an den gemeinsamen Projekten teilzunehmen. Diese Projekte sind oft Wettbewerbe, welche bestimmte Firmen ausgeschrieben haben und deren Lösung sie mit einem finanziellen Gewinn fördern.

Die beiden aktuellen Projekte von Distributed.net sind die Anwendungen „RC5-72“ und „OGR-24+“.

8.2.2.1 Projekt „RC5-72“

Das aktuell wichtigste Projekt bei Distributed.net ist der von der Firma „RSA Labs“, Anbieter von Hard- und Software im Bereich Ver- und Entschlüsselungstechnologie, ausgeschriebene „RC5-72-Wettbewerb“. Dieses Projekt koordiniert den Versuch, eine Nachricht mit Hilfe der „Brute-Force-Methode“ zu entschlüsseln, die mit dem „RC5-

---

[241] Vgl. Drum /Effiziente Methoden/ 40

[242] Die folgenden Ausführungen sind Vgl. o.V. /Provider-Distributed.net/ entnommen worden.

Verfahren“ bei 72 Bit Schlüssellänge kodiert wurde. Dazu müssen sämtliche „2^{72}“-Schlüssel systematisch durchprobiert werden.[243] Die Brute-Force-Methode ist ein Verfahren, bei dem jeder mögliche Schlüssel ausprobiert wird. „RSA Labs“ hat diesen Wettbewerb ins Leben gerufen, um zu beweisen, dass keine Verschlüsselung unknackbar ist.[244] „RC5“ ist eine Verschlüsselung von *Ronald Rivest*, die auf einer blockweisen Rotation und Vermischung der Eingangsdaten mit diversen Operationen beruht.[245] Bereits durch die erfolgreiche Entschlüsselung des Vorgängerprojektes „RC5-64“ mit Hilfe von mehr als 300.000 Teilnehmern in über 1.700 Tagen konnte bewiesen werden, dass dieser Algorithmus nicht für die Verschlüsselung wichtiger Daten geeignet ist. Auch wenn sich die Prozessorleistung in den vergangenen Jahren erheblich verbessert hat, würde das Projekt „RC5-72“ mit einem um 256mal so großen Schlüsselraum wie das „RC5-64“-Projekt bei gleichbleibender Rechenleistung der teilnehmenden Computer ca. 1200 Jahre dauern.[246]

Hierbei unterteilt ein in C++ programmierter Masterserver den Suchraum in Blöcke und vergibt diese an anfragende Server.[247] Diese unterteilen den erhaltenen Suchraum wiederum und vergeben die Blöcke an die endgültigen Clients. Der Nutzer benötigt für seine Teilnahme ähnlich wie für das Projekt SETI@home ein Client-Programm, das von Distributed.net für diverse Plattformen zur Verfügung gestellt wird.[248] Dieses muss lediglich einmal installiert und gestartet werden und läuft ab diesem Zeitpunkt als ein Hintergrundprozess auf dem Rechner.

Grundsätzlich herrscht auch beim Ausprobieren der Schlüssel zunächst einmal Unklarheit darüber, ob die Nachricht korrekt entschlüsselt wurde, da jeder Schlüssel zu irgendeiner dekodierten Nachricht führt.[249] Deshalb wurde der Beginn der geheimen Nachricht („The unknown message is:“) veröffentlicht. In diesem Zusammenhang spricht man auch von der „Known-plaintext-Attacke“. „Hätte man keinen Anhaltspunkt auf den Inhalt der Klartextmitteilung, müsste man jeden Output auf Worte oder Silben in der ent-

---

[243] Vgl. o.V. /RC5-72/

[244] Die folgenden Ausführungen sind Vgl. o.V. /Verschlüsselung/ entnommen worden.

[245] Drum /Effiziente Methoden/ 40, Vgl. o.V. /RSA Security/

[246] Vgl. o.V. /Distributed.net - Technik/

[247] Vgl. Drum /Effiziente Methoden/ 40

[248] Vgl. o.V. /Rechenzentrum/

[249] Die nachfolgenden Ausführungen sind Vgl. o.V. /RC5-72/ entnommen worden.

sprechenden Sprache prüfen und so entscheiden“[250], ob es sich tatsächlich um die gesuchte Mitteilung handelt.

Der zuvor installierte Client holt sich die zu überprüfenden Schlüssel blockweise vom Server, verarbeitet diese und vergleicht nach jedem Entschlüsselungsversuch, ob der entstandene dekodierte Text mit dem Beginn des bekannten Stücks der geheimen Nachricht übereinstimmt. Eine work unit des Clients kann aus einem oder mehreren Blöcken bestehen, wobei ein Block bei „RC5-72“ genau „2^{32}“-Schlüsselkandidaten umfasst. Die bereits abgearbeiteten Blöcke werden anschließend wieder zurück zum Server gesendet und archiviert. Dabei verhindert ein Authentifizierungsprotokoll die Ergebnisübergabe von nicht autorisierten Rechnern.[251]

8.2.2.2 Projekt „OGR-24+“

Bei dem Projekt „OGR-24+“[252] handelt es sich um eine mathematische Problemstellung; der Suche nach optimalen „Golomb-Maßstäben“.

„In der Mathematik ist der Golomb-Maßstab ein Satz von nicht negativen ganzen Zahlen, bei denen kein Paar der Zahlen die gleiche Differenz zueinander“[253] aufweist. Golomb-Maßstäbe sind nach *Dr. Solomon Golomb*, einem Professor der Mathematik mit speziellem Interesse an kombinatorischer Analysis, Zahlentheorie, Verschlüsselungstheorie und dem Kommunikationswesen, benannt. Darüber hinaus hatte er als Autor von Beiträgen der mathematischen Spiele im „Scientific American“ besondere Präferenzen für mathematische Spiele und Puzzles.[254] „Grundsätzlich ist das mit einem Lineal zu vergleichen, auf dem keiner der Abstände zwischen zwei Markierungen gleich groß ist. Ein optimaler Golomb-Maßstab ist der kürzeste Maßstab bei einer gegebenen Anzahl von Markierungen.“[255] Dabei gibt es oft mehr als einen optimalen Maßstab, wobei aber die jeweils gespiegelte Variante weggelassen wird. Die so genannten „OGRs“ spielen beispielsweise in der Radio-Astronomie, der Sensor-Platzierung für Röntgenkristallo-

---

[250] Albertin /Cracking/

[251] Vgl. Drum /Effiziente Methoden/ 40

[252] Die Projektbezeichnung OGR steht für Optimal Golomb Ruler. Vgl. o.V. /Optimaler Golomb Maßstab/

[253] o.V. /OGR-24/

[254] Vgl. o.V. /Optimaler Golomb Maßstab/

[255] o.V. /OGR-24/

graphie sowie in der Kombinatorik und der Verschlüsselungstheorie eine wichtige Rolle.

Ein Golomb-Maßstab ist eine Möglichkeit, Markierungen entlang einer Linie so zu platzieren, dass jedes der Markierungspaare einen einzigartigen Abstand misst. Wichtig ist bei diesem Verfahren, dass kein Abstand zweimal oder sogar mehrmals vorkommt.

Der in Abb. 8-3 abgebildete Golomb-Maßstab hat bei einer Länge von elf Zentimetern fünf Markierungen und zeigt bis auf den Abstand von sechs Zentimetern alle Abstände lediglich einmal auf Die Zahlen unter den Markierungen zeigen den Abstand von der linken Kante aus.

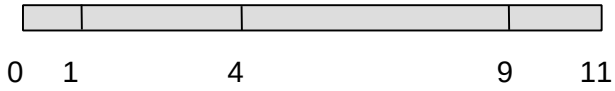

Abb. 8-3: Golomb-Maßstab mit fünf Markierungen[256]

Die Erstellung einer Tabelle (Vgl. Abb. 8-4) mit allen Markierungspaaren und ihren zugehörigen Abständen, vereinfacht die Überprüfung, ob der obige Maßstab wirklich Golomb ist.[257]

Markierung 1	Markierung 2	Abstand
0	1	1
0	4	4
0	9	9
0	11	11
1	4	3
1	9	8
1	11	10
4	9	5
4	11	7
9	11	2

Abb. 8-4: Abstände im Golomb-Maßstab[258]

In der dritten Spalte stehen keine doppelten Abstände, denn Golomb-Maßstäbe müssen nicht jeden Abstand messen, sondern nur unterschiedliche. „Das Ziel der Optimierung

[256] Vgl. o.V. /OGR-Project/

[257] Vgl. o.V. /Optimaler Golomb Maßstab/

[258] Vgl. o.V. /OGR-Project/

von Golomb-Maßstäben liegt darin, sie so kurz wie möglich zu machen, ohne irgendwelche Abstände zu duplizieren.“[259]

Der Aufwand für das Finden und Beweisen von OGR steigt mit der Anzahl der Markierungen exponentiell an und eignet sich aufgrund des zum Teil sehr hohen Rechenaufwandes gut als eine Problemstellung für das verteilte Rechnen.[260] Wichtig ist vor allem, dass grundsätzlich der gesamte Möglichkeitsraum durchgerechnet werden muss, da immer eine kürzere Variante gefunden werden kann.

8.2.3 WebOS

Während das World Wide Web einen relativ einfachen Zugriff auf geographisch verteilte Daten ermöglicht, gestaltet sich der Zugriff auf geographisch verteilte Computerressourcen weiterhin schwierig.[261] Deshalb erfordern weit verteilte Anwendungen Zugriff auf verteilte Prozessoren, Arbeitsspeicher sowie Festplattenspeicher in anwendungsspezifischer Art und Weise. Beispielsweise sind bekannte Dienste wie „Digital´s Alta Vista“ oder „Netscape´s“-Downloadservice geographisch nachgebildet, um die Bandbreite zu erhöhen, die Latenzzeiten zu reduzieren und die Verfügbarkeit zu verbessern; denn keine einzelne Internetverbindung ist in der Lage mehrere Millionen Internetnutzer zu versorgen. Eine Lösungsmöglichkeit für dieses Problem stellt „WebOS“ als ein Framework zur Unterstützung geographisch verteilter, hoch verfügbarer, schrittweise erweiterbarer und dynamisch rekonfigurierbarer Anwendungen dar. Dieses Framework beinhaltet Mechanismen für globale Benennung, dauerhafte Speicherung, Prozessausführung, Ressourcenmanagement, Zertifizierung und Sicherheit. Das „WebOS-Framework“ ermöglicht ein neues Paradigma für Internetdienste, indem Teile ihrer dynamischen Funktionalität per Internet angeboten werden können. Diese dynamisch rekonfigurierbaren und geographisch mobilen Dienste besitzen eine Reihe von Vorteilen, welche neben einer besseren „end-to-end“-Verfügbarkeit, eine gute Kosteneffizienz sowie ein verringertes Ausfallrisiko beinhalten. Das Ziel von WebOS ist die Bereitstellung einer Struktur, um Anwendungsentwickler bei der Benutzung von programmierbaren, aktiven Netzwerkkomponenten zu unterstützen.

---

[259] o.V. /Optimaler Golomb Maßstab/

[260] Die nachfolgenden Ausführungen sind Vgl. Albertin /Cracking/ entnommen worden.

[261] Die folgenden Ausführungen sind Vgl. Vahdat et al /WebOS/ entnommen worden.

8.2.3.1 Architektur

Die WebOS-Architektur besteht aus verschiedenen Komponenten, welche die Dienste zwischen lokalen und verteilten Betriebssystemen durch die Nutzung geographisch verteilter Ressourcen vereinfachen.[262] Diese Komponenten sind „Global Naming“, „Wide-Area File System“, „Security and Authentication“ und „Process Control”.

- Global Naming: Viele weit verbreitete Dienste sind geographisch verteilt. Um eine gute Systemperformance bieten zu können, muss ein Client-Programm in der Lage sein, den Server dynamisch zu lokalisieren. In WebOS übernimmt das „Global Naming“ die Zuordnung der Servicenamen zu multiplen Servern und die Ausfallsicherung bei plötzlich auftretenden Serverproblemen. Diese Operationen werden durch so genannte „Smart Clients“ durchgeführt, welche die spezifischen Dienste für die Client-Maschine flexibel erweitern. Abb. 8-5 gibt einen Überblick über die „Smart-Client“-Architektur.

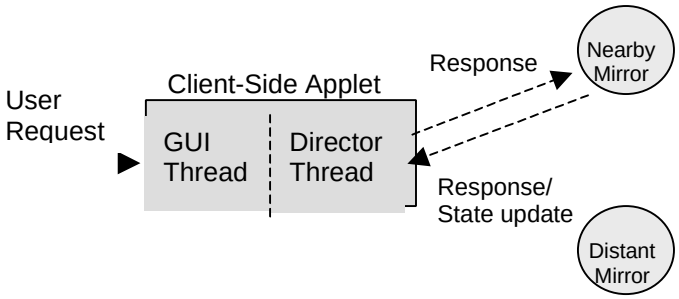

Abb. 8-5: Smart Client Architektur[263]

Diese Smart Client Architektur besteht aus zwei kooperierenden Threads, dem „GUI Thread“ und dem „Director Thread“. Während der „GUI Thread“ als graphisches Interface die Sicht der Nutzer auf die Dienste repräsentiert und Anfragen der Nutzer passieren lässt, ist der „Director Thread“ für den Service Provider verantwortlich und die damit in Verbindung stehende Bereitstellung des bestmöglichen Services für die Nutzer.

---

[262] Die folgenden Ausführungen sind Vgl. Vahdat et.al /WebOS/ entnommen worden.

[263] Vahdat et al /WebOS/

- Wide-Area File System: WebFS bietet ein globales, konsistentes Dateisystem auf der Grundlage des Namensraumes von HTTP zur Unterstützung der Replikation und des „wide-scale-sharing“.

- Security and Authentication: Um die Anwendungen über organisatorische Grenzen hinweg unterstützen zu können, wurde in WebOS ein Modell zur Gewährleistung der Sicherheit entwickelt.

- Process Control: Die in WebOS auszuführenden Prozesse auf entfernten Rechnern sollten so einfach wie entsprechende Operationen auf lokalen Rechnern gestaltet sein. Das dafür zugrunde liegende System ist zum einen für die Authentifizierung des Aufrufers und zum anderen für die Überwachung der Prozessabläufe verantwortlich.

8.2.3.2 Anwendungen von WebOS

Für die spezielle Nutzung des WebOS-Frameworks wurden vier Anwendungen entwickelt. Während der Entwicklungsprozess für die Applikationen „Internet Chat“ und „Remote Compute Engine“ bereits weitestgehend abgeschlossen ist, steht dies für die Anwendungsgbiete „Wide Area Cooperative Cache“ sowie „Internet Weather“ noch bevor.[264]

- Internet Chat: Der Internet Chat ermöglicht es Nutzern so genannte „chat rooms“ zu betreten und zu verlassen, um mit anderen, zur gleichen Zeit in einem logischen Raum anwesenden, Personen zu kommunizieren. In WebOS können diese mit Hilfe des Smart Client erreicht werden. Dabei wählen die Smart Clients den am wenigsten genutzten Server, um einen Lastenausgleich zu erreichen. Um den durch das WebOS-Framework möglichen Nutzen zu erhöhen, wurden zwei Versionen der „Internet Chat“-Anwendung, auf der Basis identischer Semantik, implementiert. Nachdem die ursprüngliche Implementierung auf 1200 Zeilen Java-Programmcode und 4200 Zeilen der Programmiersprache C++ basierte, reduzierte WebOS den Java-Code auf nur noch 850 Zeilen und ersetzte die 4200 Zeilen C++ komplett durch einen spezifischen „Chat Server Code“. Der Hauptgrund dieser Komplexitätsvereinfachung bestand in der Ersetzung eines separaten Codes für das Management der Kommunikation und der durchgehenden Archivierung der im Chat ausgetauschten Inhalte.

---

[264] Die Ausführungen des folgenden Kapitels sind Vahdat et al /WebOS/ entnommen worden.

- Remote Compute Engine: Bei der Nutzung von WebOS wird geographisch entfernten Nutzern ermöglicht, lokale Programme aufzurufen und auf die gleichen Daten wie lokale Nutzer zuzugreifen. Darüber hinaus bearbeitet WebOS eine Vielzahl von Diensten wie zum Beispiel die Authentifizierung von Nutzern, die Sicherheitsgewährleistung bei Veröffentlichung von privaten Dateien und die Bereitstellung eines Umfeldes, welches die lokalen Programme vor möglichen Angriffen nicht autorisierter Benutzer schützt.

- Wide Area Cooperative Cache: WebOS dient der Schaffung eines geographisch verteilten so genannten „Web-cooperative-Cache“, um durch eine intelligentere Art des Zwischenspeicherns von Webinhalten einen direkten Vorteil für das Internet zu erzielen. Gegenwärtig bestehende Ansätze des hierarchischen Zwischenspeicherns sind aufgrund technischer Unzulänglichkeiten und des enormen Wachstum sowohl der Speicherkapazitäten als auch der Verarbeitungsgeschwindigkeit sehr kritisch anzusehen. WebOS vereinfacht die Implementierung dieses „Web-cooperative-Cache“ in verschiedener Art und Weise: So werden Smart Clients genutzt, um den entsprechenden „Proxy-Cache“ anzusprechen und Speicherdateien zwischen den jeweiligen „Proxies“ zu transportieren

- Internet Weather: Eine Vielzahl von Internetseiten versuchen gegenwärtig regelmäßig aktualisierte Informationen in den Bereichen Wetter, Verkehr und anderen anzubieten. Diese Informationen können für strategische Standortentscheidungen sehr wertvoll sein. In der Implementierung von WebOS wird ein zentraler Server genutzt, der die „Smart Client“- Anwendungen für potentielle Nutzer dieser Dienste unterstützt.

Die Zielsetzung dieses Projektes war es, die entsprechenden Systemabstraktionen, Mechanismen und grundlegenden Richtlinien zu verstehen, die für die aufkommende Klasse weit verteilter Anwendungen angemessen sind.[265] Das Forschungsprojekt wurde inzwischen weitestgehend abgeschlossen. Einige der zugrunde liegenden Ideen fanden im Rahmen des Unternehmens „webos.com“ kommerzielle Anwendung.

8.2.4 Legion

Die zuvor betrachteten Systeme SETI@home und Distributed.net verfügen nicht über eine vermittelnde Schicht zwischen Hardware und Applikation. Dies hat zur Folge, dass

[265] Die Ausführungen dieses Abschnitts sind Vgl. Vahdat /Status of WebOS/ entnommen worden.

die Applikation für jedes zu bearbeitende Problem direkt auf der Infrastruktur aufgesetzt werden muss und somit für jedes Projekt eine Anpassung erforderlich ist.

„Legion“ ist eine das Grid Computing unterstützende Software Architektur.[266] Darunter versteht man ein verteiltes System, welches aus einer Vielzahl geographisch weit verteilter Rechner entsteht, die zu verschiedenen Organisationen gehören, sich dem Systembenutzer jedoch wie ein lokaler Rechner mit gewaltigen Ressourcen und enormer Rechenfähigkeit präsentiert. Dank der erfolgreichen Entwicklung der Netzverbindungen, die eine Datenübertragung bis hin zu einigen Terabits pro Sekunde ermöglichen, erhielten die Bestrebungen nach einer einzigen virtuellen Maschine in den letzten Jahren kontinuierlich realistischere Züge.

Das im Jahre 1993 vom Department of Computer Science der Universität von Virginia und mehreren Partnern gegründete Legion-Projekt folgt dieser Vision und versucht eine Middleware aufzubauen, die alle Ressourcen umfassen und koordinieren kann. Es wird in der Literatur als das wohl am meisten ausgereifte Metacomputing-System bezeichnet und ist bereits auf mehreren Testumgebungen installiert. Das Legion-System bietet auf verschiedenen Plattformen transparenten Zugang zu verteilten Ressourcen an, die zusammen einen virtuellen Höchstleistungsrechner darstellen.

Diese verteilten Ressourcen können aus Rechnern, Bibliotheken, Simulationen, Kameras u.ä. bestehen. Auf ihnen werden Dienste wie eine Verteilungsstrategie, Datenmanagement, Fehlertoleranz und Sicherheitsmechanismen angeboten. Das jeweils auf dem lokalen Betriebssystem aufgesetzte Legion- Laufzeitsystem, LRTL - Legion Runtime Library,[267] vermittelt dabei zwischen den von der Applikation benötigten Ressourcen und sorgt darüber hinaus für die Einhaltung der lokalen Sicherheitsrichtlinien.

Von den in Kapitel 7.2.2 beschriebenen Anforderungen an Metacomputing-Systeme hat sich Legion die nachfolgend aufgeführten zum Ziel erklärt:[268]

- Fehlertoleranz: Die Verknüpfung vieler Rechner zu einem System impliziert die Fehlerhäufigkeit durch Maschinenversagen, Verbindungsabbruch etc. Legion muss deshalb in der Lage sein, die Zustände der Ressourcen zu überwachen und während des

---

[266] Die folgenden Ausführungen sind Vgl. o.V. /Architektur/, Vgl. Drum /Effiziente Methoden/ 48 entnommen worden.

[267] Vgl. Eickerman et al. /Metacomputing in gigabit environments/ 1847 ff.

[268] Die folgenden Ausführungen sind Vgl. Schröder /Grid Computing/, Vgl. Schurz /Gridcomputing/, Vgl. o.V. /Architektur/ entnommen worden.

Betriebs ausfallende Ressourcen durch dynamische Rekonfiguration vor dem Benutzer und der Anwendung verbergen.

- Heterogenität: Legion sollte die Interoperabilität und Kompatibilität zwischen den Hardware- und Softwarekomponenten unterstützen.

- Institutionsautonomie: Legion versucht verschiedene Ressourcen zu integrieren und zu koordinieren, um die Leistungsfähigkeit zu steigern; besitzt diese Ressourcen jedoch nicht. Die zur Verfügung gestellten Ressourcen bleiben weiterhin unter lokaler Kontrolle der entsprechenden Organisationen. Durch dieses Vorgehen soll einerseits eine individuelle Kontrolle der Ressourcen ermöglicht werden und andererseits jeder Benutzer in der Lage sein, spezifizieren zu können, wie viel einer bestimmten Ressource benutzt werden darf, wann diese benutzt werden darf, wer Zugriff auf diese Ressourcen haben soll und wer nicht.

- Transparenz: Legion muss in der Lage sein, allgemeine Benutzer anzusprechen und eine einfache Schnittstelle zu Diensten anzubieten, um die Benutzer vor der Komplexität der Hardwareumgebung und der Kommunikationsprozesse abzuschirmen.

- Performance: Zur Erreichung einer angemessenen Performance sollte Legion einfach benutzbare parallele Prozesse unterstützen und einen hohen Grad an Parallelität ermöglichen.

- Erweiterbarkeit: Ein Metasystem wie Legion muss flexibel gestaltet sein, um den Ansprüchen, sowohl der Anwendungen als auch der Benutzer genügen zu können. Um alle Komponenten und Dienste erweitern und ersetzen zu können, wurde Legion nach objektorientiertem Paradigma entworfen und implementiert.

- Sicherheit: Die Gewährleistung der Sicherheit in einem Metacomputer ist ein sehr anspruchsvolles Ziel, für dessen Umsetzung verschiedene Ressourcen und eine autonome Verwaltung benötigt werden. Deshalb sollte es die Aufgabe von Legion sein, bereits bestehende Sicherheitskonzepte nicht zu schwächen, sondern vielmehr Sicherheitsmechanismen bereit zu stellen und Benutzern die Möglichkeit zu geben, ihre Sicherheitsansprüche individuell zu befriedigen. Damit stellt die Sicherheitsinfrastruktur ein überaus wichtiges Element für den Erfolg oder den Misserfolg einer Metacomputing-Umgebung dar.

- Skalierbarkeit: Die hohe Anzahl beteiligter Rechner bedeutet, dass zu jedem Zeitpunkt beliebig viele Rechner an Legion teilnehmen können, es aber auch beliebig viele verlassen können. Aus diesem Grund muss eine skalierbare Architektur gewährleistet werden, in der keine zentrale Ressourcenverwaltung existiert, sondern das System selbst vollkommen verteilt ist.

- Eindeutiger Namensraum: Legion benutzt einen eindeutigen, persistenten Namensraum (context space) für den Zugriff auf Dateien und Daten, damit alle Objekte, unter Einhaltung entsprechender Sicherheitsbedingungen, mit anderen Objekten interagieren können ohne geographische oder logische Grenzen berücksichtigen zu müssen.

Grimshaw definiert Legion als „a reflective object-based wide-area system designed from first principles to address the issues and to thus simplify the task of writing distributed applications“[269]. „Legion ist objektorientiert, wobei die Basisobjekte sowie Basisfunktionalitäten von Objekten wie Erstellung, Löschung und Aktivierung vorgegeben sind.“[270] Zur Leistungssteigerung von Anwendungen bietet Legion zwei Möglichkeiten an: Zum einen soll sichergestellt werden, dass für serielle Programme der günstigste Knoten zur Berechnung gefunden wird und zum anderen wird die parallele Ausführung unterstützt. Um dies realisieren zu können, werden von Legion-Bibliotheken die Funktionalitäten von Parallel Virtual Machines nachgebildet. Legion selbst ist in der Programmiersprache „Mentat Programming Language“, kurz MPL[271], geschrieben.

8.2.4.1 Architektur und Funktionsweise

Um die bereits erläuterten Anforderungen erfüllen zu können, ist die Eigenschaft der Flexibilität für Legion unerlässlich. Das Objektmodell mit den Mechanismen der Vererbung und Kapselung ist eine ideale Möglichkeit diese Anforderungen zu realisieren.[272] Legion ist in Klassen und Metaklassen[273] organisiert, wobei einzelne Objekte wiederum Instanzen von Klassen darstellen. Demzufolge werden alle Ressourcen wie Rechner,

---

[269] Grimshaw /Legion/

[270] Drum /Effiziente Methoden/ 49

[271] Die MPL ist eine parallele C++ Programmiersprache, die an der Universität von Virginia entwickelt wurde und vorrangig für die plattformübergreifende parallele Verarbeitung eingesetzt wird. Vgl. o.V. /Architektur/

[272] Vgl. o.V. /Architektur/

[273] Metaklassen werden als Klassen von Klassen definiert und haben eine Art Managerfunktion. Vgl. Schurz /Gridcomputing/

Speicher, Dateien etc. als unabhängige Objekte dargestellt, welche durch gegenseitige, asynchrone Methodenaufrufe miteinander kommunizieren.[274] Hierfür werden aus ortsunabhängigen Identifikatoren auf der Anwenderseite, den so genannten „Legion Object Identifier - LOID“, ortsabhängige Identifikatoren, auch als „Legion Object Address - LOA“ bezeichnet, generiert, über die die Applikation auf die Objekte zugreifen kann. Diese verfügen sowohl über die physikalische Adresse des Objektes, als auch über einen zugehörigen RSA-Schlüssel und ein Zertifikat nach X.509[275]. Abb. 8-6 zeigt die einzelnen Bestandteile eines Legion Object Identifier.

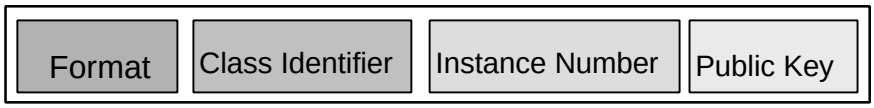

Abb. 8-6: Format einer LOID[276]

Jedes dieser Objekte enthält alle Informationen über die repräsentierte Ressource wie z.B. Prozessortyp, Speicherkapazität, Objektadresse, Sicherheitsrichtlinie etc. Die große Anzahl möglicher Objekte innerhalb des Systems ermöglicht es, Objekte automatisch in einem persistenten Zustand auf permanentem Speicher auszulagern. Diese ausgelagerten und somit inaktiven Objekte können durch so genannte OPA[277]-Objekte lokalisiert und bei Bedarf in einen aktiven Zustand versetzt werden.

In Legion existieren vordefinierte „Kernobjekte und zugehörige Implementierungen, die für die Kommunikation zuständig sind.“[278] Während die Kernobjekte von den Benutzern auf ihren Rechnern installiert werden müssen, um an Legion teilnehmen zu können, ist die Benutzung der zugehörigen Implementierungen unverbindlich. Demnach können die Benutzer die Implementierungen nach eigenem Bedarf modifizieren oder durch eigene Implementierungen ersetzen. Die von Legion definierten Kernobjekte, in der Fachliteratur auch als „Core Objects“ bezeichnet, lassen sich in „Klassen und Meta-

[274] Die nachfolgenden Ausführungen sind Vgl. o.V. /Architektur/, Vgl. Drum /Effiziente Methoden/ 49 entnommen worden.

[275] Vgl. o.V. /X.509-Zertifikat/

[276] Drum /Effiziente Methoden/ 49

[277] OPA - Object Persistent Address

[278] o.V. /Architektur/

klassen“, in „Hostobjekte“, in „Vaultobjekte“, in „Implementierungsobjekte“, in „Bindungsagenten“ sowie in „Kontextobjekte und Kontextraum“ gliedern.[279]

Klassenobjekte sind dafür zuständig, Objekte zu definieren, zu verwalten, zu instanziieren, diese bei Bedarf zu aktivieren beziehungsweise zu deaktivieren sowie die Kommunikationsbindungen zu erstellen.

Hostobjekte (Host Objects) repräsentieren Rechenressourcen in Legion, d.h. einen einzelnen Prozessor oder eine Aggregation von Prozessoren. Hostobjekte sind dafür verantwortlich, Objekte zum Laufen zu bringen, Lauffehler zu überwachen sowie zu entscheiden, welche Rechte den Objekten zugewiesen werden. Sie stellen somit eine wichtige Säule für die Sicherheit dar.

Vaultobjekte (Vaults) erzeugen für jedes inaktive Objekt eine „Object Persistent Representation“, kurz OPR, speichern diese im persistenten Speicher ab und verwalten sie. Wird ein Objekt wieder aktiviert, leitet das Vaultobjekt die dem Objekt zugehörige OPR zum Hostobjekt. Nach einem ähnlichen Mechanismus stellen Vaultobjekte wesentliche Elemente für die Migration dar. Sobald ein Objekt aus Gründen wie z.B. der Prozessunterbrechung von einem Hostobjekt zu einem anderen Hostobjekt verlegt wird, muss die entsprechende OPR des Objektes ebenfalls zu dem neuen Vaultobjekt transportiert werden.

Implementierungsobjekte (Implementation Objects) verfügen über Methoden wie Lesen und Schreiben, ähnlich dem normalen Dateiobjekt konventioneller Betriebssysteme. Der grundsätzliche Unterschied zwischen normalen Dateiobjekten und Implementierungsobjekten besteht darin, dass letztere nicht uneingeschränkt gelesen und geschrieben werden dürfen. Sobald ein Implementierungsobjekt gelesen ist, darf es nicht mehr geschrieben werden, sondern bleibt konstant bis es gelöscht ist. Aufgrund der Anforderung der Heterogenität der Hardware- und Softwareumgebung haben Objekte in Legion üblicherweise mehrere Implementierungen, die im Klassenobjekt gespeichert sind.

Bindungsagenten (Binding agents) unterstützen den Bindungsprozess zwischen dem Legion Object Idenifier (Vgl. Abb. 8-6) und der Objektadresse. Unter technischen Gesichtspunkten sind diese nicht unbedingt notwendig, zumal ein Client zur Erstellung einer Bindung auch direkt die Klassenhierarchie durchsuchen könnte. Da dieses Vorgehen

---

[279] Die nachfolgenden Ausführungen sind Vgl. Schurz /Gridcomputing/, o.V. /Architektur/ entnommen worden.

jedoch sehr zeitintensiv ist, sind Bindungsagenten auf den Bindungsprozess selbst spezialisiert und funktionieren ähnlich einer Datenbank, die das Suchergebnis im Cache speichert, um den Suchprozess zu beschleunigen.
Ein Kontextobjekt ist ein Namensraum. Dieser entsteht aus einer Menge von Kontextobjekten, die einen gerichteten Graph bilden. Mittels dieses Graphen organisiert Legion die Informationen der Ressourcen, wobei jedes Kontextobjekt seinen Verwaltungsbereich hat. Je nachdem in welchem Bereich sich das entsprechende Objekt befindet, bekommt es die LOID ihres Kontextobjektes zugewiesen.
Die in Legion definierten Basisklassen spezifizieren lediglich die Funktionalität und die Schnittstellen der Basisobjekte, die von den Benutzern sowohl erweiterbar als auch ersetzbar sind.[280] Die Verteilung der Prozesse innerhalb von Legion findet in fünf aufeinander folgenden Schritten statt.
Zunächst werden alle Ressourcen wie Hostobjekte, Vaultobjekte u.a. in einer Ressourcendatenbank gesammelt. Im Anschluss daran stellt der für die Prozessverteilung zuständige Prozess, auch als „Scheduler“ bezeichnet, Anfragen an die Datenbank nach Möglichkeiten, bestimmte Anforderungen zu erfüllen. Aus den von der Datenbank gelieferten Ergebnissen werden der Anwendung mehrere Prozessverteilungsalternativen angeboten. Dabei kann die Anwendung selbst oder der Prozess, der für die Anwendung die Prozessverteilung übernimmt, mit den Objekten der Ressourcen kommunizieren und diese belegen. Diese Information wird von der Prozessverteilung zur Datenbank weitergeleitet.
Zur zusätzlichen Leistungssteigerung kann die Prozessverteilung auch vom Benutzer oder innerhalb der Anwendung implementiert werden. Anhand der Richtlinien der entsprechenden Ressource kann innerhalb von Legion festgestellt werden, ob eine Ressource genutzt werden kann und darf. Damit wird dem Benutzer der Ressource die Autonomie gewährleistet.

8.2.4.2 Nutzungsstatus

Im Juni 1996 wurde mit der Programmierung der Version „VaL 1.0“[281] begonnen, die als komplett lauffähige Implementierung bereits im Dezember 1997 veröffentlicht wur-

---

[280] Die nachfolgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 50 entnommen worden.

[281] Die im Rahmen der ersten Version verwendete Abkürzung „VaL“ steht für Virginia Legion. Vgl. o.V. /Architektur/

de.[282] Für den Einsatz von Legion für global verteilte Anwendungen gibt es eine Reihe von Anwendungsklassen.[283]

Eine überaus wertvolle Form der Nutzung von Legion ist sowohl der Einsatz in allgemeinen Anwendungen, die wenig kommunizieren, wie von der Universität von Virginia bereits für die Modellierung neuronaler Netzwerke implementiert, als auch der Einsatz in mehreren verschiedenen Anwendungen, die miteinander kooperieren. Darüber hinaus können Anwendungen, die einen großen Suchraum unabhängig voneinander bearbeiten, ebenfalls Vorteile einer Legion-Umgebung nutzen. In diesem Fall ist lediglich ein Prozessverteilungsmechanismus notwendig, der die Aufgaben über die Ressourcen von Legion verteilt. Von der objektorientierten verteilten Umgebung und der Ressourcenverwaltung eines Legion-Systems profitieren vor allem einfache entfernte Ausführungen zusammen mit der Möglichkeit, die entfernten Programme mittels eines Werkzeuges zuvor zu übersetzen

Im Januar 2001 wurden die Rechte an Legion an das Unternehmen „Avaki“ verkauft, mit dem Ziel, die Technologie von Legion so zu kommerzialisieren, dass die Bedürfnisse der potentiellen Verbraucher und Ressourcenanbieter angesprochen werden.[284] Unabhängig von Erfolg oder Misserfolg wird die Forschung und Entwicklung von Legion an der Universität von Virginia fortgesetzt. Trotz bedeutungsvoller Fortschritte hinsichtlich der Nutzung von verteilten Ressourcen besteht auch zukünftig ein erheblicher Forschungsbedarf in Bereichen wie Sicherheit, Fehlertoleranz und Interoperabilität mit anderen Grid-Infrastrukturen. Während „der Entwicklung von Legion haben sich mehrere Institute und Firmen zu sogenannten TestBeds zusammengeschlossen. Innerhalb dieser TestBeds finden Anwendungen durch die Legion Infrastruktur eine Metacomputing-Umgebung vor.“[285] So wurden Testumgebungen wie beispielsweise das „Vanet TestBed“ als eine Metacomputing-Umgebung mit Hilfe von Institutsrechnern an der Universität von Virginia als auch das „Centurio TestBed“ mit einer Konfiguration von 384 Prozessoren[286] und einer Spitzenleistung von über 240 GigaFLOP pro Sekunde rea-

---

[282] Vgl. o.V. /Architektur/

[283] Die Ausführungen des folgenden Abschnitts sind Vgl. Drum /Effiziente Methoden/ 51,52 entnommen worden.

[284] Vgl. o.V. /Architektur/

[285] Drum /Effiziente Methoden/ 52

[286] Diese 384 Prozessoren setzen sich aus 256 Pentium II Rechnern mit einer Taktfrequenz von 400MHz und 128 DEC Alpha Rechnern mit einer Taktfrequenz von 533MHz zusammen. Vgl. Drum /Effiziente Methoden/ 52

lisiert. Auf dieser Konfiguration laufen zum Beispiel Wetter-Modellierungen, Simulationen zu chemischen Gasphasenabscheidungs-prozessen und einige mehr.

9 Sicherheitsaspekte in Metacomputing-Systemen

Der Aspekt der Computer- und Softwaresicherheit ist ein enorm wichtiges Thema im Bereich des wissenschaftlich-technischen Rechnens und wird in naher Zukunft kontinuierlich an Bedeutung zunehmen. Der Begriff der Computersicherheit wird in der Fachliteratur oft synonym mit den Begriffen „Computer Security“ und „DV-Sicherheit“ verwendet. „Unter Computersicherheit versteht man, dass Maßnahmen getroffen werden, die sicherstellen, dass nur befugte Personen Zugriff auf ein Computersystem haben. Da auf Computersystemen immer wertvollere Daten gespeichert werden, wächst die Wichtigkeit der Computersicherheit.“[287] Unter unbefugte Nutzung fallen sowohl die Bedrohungen durch „Spionage“ als unberechtigte Kenntnisnahme von Objekten („unauthorized disclosure“) als auch „Sabotage“ als unberechtigte Änderung („integrity“) beziehungsweise Überlastung von Objekten („denial of service“).[288] Die zu schützenden Komponenten sind neben Daten und Kommunikationswegen auch Ressourcen, wobei für jede der drei Komponenten Vertraulichkeit, Integrität und Verfügbarkeit zu gewährleisten sind.[289]

Im Zusammenhang mit der Computersicherheit versteht man unter Vertraulichkeit die Eigenschaft, Informationen nur berechtigten Subjekten zugänglich zu machen.[290] Demnach bedeutet ein unbefugter Informationsgewinn gleichzeitig einen Verlust der Vertraulichkeit.

Die Integrität hingegen charakterisiert den Fall, dass keine unbefugten Modifikationen an den Daten vorgenommen werden können. Werden Informationen von einem unberechtigten Subjekt verändert, geht die Integrität dieser Informationen verloren.

Die Verfügbarkeit der vollständigen Funktionalität eines Computersystems kann als eine eigenständige Eigenschaft eines Computersystems angesehen werden. Wenn es einem Angreifer gelingt, ein System so zu überlasten, dass es zu einer spürbaren Beein-

---

[287] o.V. /Computersicherheit/ 3

[288] Vgl. Murauer /Zugriffschutz in Computersystemen/

[289] Vgl. Drum /Effiziente Methoden/ 34

[290] Die folgenden Ausführungen sind Murauer /Zugriffschutz in Computersystemen/ 4 ff. entnommen worden.

trächtigung kommt, dann spricht man vom Verlust der Verfügbarkeit. Bekannte Beispiele, die auf die Verfügbarkeit von Computersystemen abzielen, sind Computerviren.
Der Aspekt der Computersicherheit gewinnt im Rahmen des Metacomputing vorwiegend durch die Wechselwirkung von Bedeutung und Entwurf des Netzes als auch der zunehmenden Kommerzialisierung der Netzinfrastruktur kontinuierlich an Einfluss. Dies erfordert bei Anwendungen des wissenschaftlich-technischen Rechnens, das im Allgemeinen auf hohe Rechenleistung optimiert ist, ein Abwägen zwischen gewonnener Leistung und notwendiger Sicherheit.[291]
Die Projekte SETI@home und Distributed.net, auch als einfache Metacomputing-Systeme zum wissenschaftlich-technischen Rechnen bezeichnet, erfreuen sich mit ihren jeweiligen Forschungsinhalten einer regen Teilnahme der Internetnutzer. Was für viele Teilnehmer auf den ersten Blick eher unwichtig erscheint, kann sich, durch den für die Teilnahme an den entsprechenden Projekten notwendigen Download der Client-Software, im Nachhinein als nachteilig erweisen. Wenngleich die verfolgten Ziele dieser Projekte hoch interessant sind, ist die verwendete Methode, dass die Software noch nicht vollständig im Quelltext erhältlich ist, sehr kritisch zu sehen. Bei Distributed.net verlangt die Client-Software beispielsweise nach erfolgter Installation mit, für den Nutzer, unbekannten Servern zu kommunizieren. Diese sind in der Lage, an den meisten Firewalls und anderen Sicherheitssystemen vorbei Daten zu übertragen, ohne die Möglichkeit zu haben, nachsehen zu können, was dieses Programm eigentlich macht.[292] Darüber hinaus besitzt diese Software die Möglichkeit, sich in einem Windows-Betriebssystem mehr oder weniger wirkungsvoll zu verstecken. Es wäre zum Beispiel denkbar, dass das Programm, vom Benutzer unbemerkt, Informationen aus dem Rechner „herausschafft“ und an Dritte weitergibt. Unter dieser Voraussetzung lässt sich das Sicherheitsargument der Nichtfreigabe des Quelltextes von Distributed.net nicht mehr unterstützen.[293] Dabei soll aus Sicherheitsgründen weniger der Abbruch des Projekt das Ziel sein als vielmehr die uneingeschränkte Freigabe des Quelltextes.
In WebOS ist eine Sicherheitsarchitektur, das so genannte „CRISIS“, für die weit verteilte Authentifizierung und Zugriffskontrolle verantwortlich.[294] Sie stellt ein Subsystem

---

[291] Vgl. Drum /Effiziente Methoden/ 34

[292] Vgl. o.V. /Sicherheit in Distributed.net/

[293] Vgl. o.V. /Sicherheit in Distributed.net/

[294] Belani et al. /CRISIS Architecture/

von WebOS dar, welches die Funktionalität von Betriebssystemen um die Aspekte Sicherheit, Ressourcenmanagement und andere erweitert.

Das Legion-Projekt definierte als eines der Entwicklungsziele, „die Erhaltung der Sicherheit auf den beteiligten Knoten für deren Besitzer und die Sicherheit der Daten innerhalb einer Anwendung für den Benutzer.“[295] Die Tatsache, dass die an Legion beteiligten Rechner zu verschiedenen Netzdomänen oder Organisationen gehören können und es keinen Systembesitzer oder Systemadministrator mit höchsten Privilegien gibt, erfordert ein Sicherheitsmodell, dass die Autonomie der beteiligten Systeme bewahrt, keine Lücken offenbart und den Sicherheitsbedarf der verschiedenen Benutzer und Ressourceninhaber berücksichtigt.[296] In Legion kann jedes Objekt seine eigenen Sicherheitsmaßnahmen festlegen, wobei der Eigentümer eines Objektes fälschungssichere Berechtigungen vergibt, die bei einem Methodenaufruf übergeben und geprüft werden.[297] Werden Objekte indirekt über eine Kette von Aufrufen angesprochen, können dem Aufruf Berechtigungslisten, so genannte „Credentials“, mitgegeben werden.[298] Diese Listen werden bei allen folgenden Aufrufen weitergeleitet, so dass das Zielobjekt die Prüfung und Berechtigung vornehmen kann. Jedes Objekt in Legion beinhaltet eine Liste der eigenen Rechte und der Methodenaufrufe, die es erlauben kann. Zur Vereinfachung der Überprüfung der entsprechenden Rechte, kann die „MayI“-Methode als Klassenmitgliedsfunktion implementiert werden. Diese wird in jeder Klasse obligatorisch eingebaut, um den Benutzern die Möglichkeit zu bieten, ihre eigene Sicherheitsrichtlinie zu realisieren.[299] Bei jedem Methodenaufruf aktiviert Legion automatisch die MayI-Funktion und führt die Sicherheitsprüfung durch. Abhängig vom Prüfergebnis wird entweder die aufgerufene Methode ausgeführt oder der Aufruf abgesagt. Anhand dieses Vorgehens können die geforderten Sicherheitsrichtlinien in Legion-Systemen weitestgehend verwirklicht werden. Darüber hinaus muss die physische Sicherheit des Rechnersystems, die Hardware, einer ständigen Sicherheitsüberprüfung unterliegen, damit der Legion-Code auf dem jeweiligen Rechner normal und richtig funktioniert.[300]

---

[295] Drum /Effiziente Methoden/ 51

[296] Vgl. o.V. /Architektur/

[297] Vgl. o.V. /Architektur/

[298] Vgl. Drum /Effiziente Methoden/ 51

[299] Vgl. o.V. /Architektur/

[300] Vgl. o.V. /Architektur/

Der Sicherheitsaspekt stellt einen sehr wichtigen Faktor für eine Metacomputing-Umgebung dar und ist maßgeblich am Erfolg oder Misserfolg eines Metacomputing-Systems beteiligt.

10 Fazit

In dieser Arbeit wurde ausgehend von den Grundlagen und der Abgrenzung zwischen Parallelem und Verteiltem Rechnen auf die Technologie des Metacomputing sowie auf die Anforderungen an Metacomputing-Systeme eingegangen. In diesem Zusammenhang wurden wesentliche Technologien zum wissenschaftlich-technischen Rechnen, so genannte Metacomputing-Infrastrukturen, analysiert.

„In dem sich rasant entwickelnden Gebiet des Metacomputing kann eine abschließende Bewertung nur unvollständig sein. Der Stand der Forschung konzentriert sich zur Zeit nicht in einer homogenen Umgebung, sondern in einer Spezialisierung der Dienste, deren Kombination über eine dezentrale Verwaltung den Metacomputer bildet, dabei koordiniert die dezentrale Verwaltung die untereinander kommunizierenden Dienste.“[301]

Der Hauptvorteil von Metacomputing-Architekturen ist in der traditionell im Vordergrund stehenden Geschwindigkeitssteigerung zu sehen. „Diese wird durch die parallele Ausführung der Anwendung und durch die Wahl eines leistungsstärkeren Zielsystems erreicht.“[302] Darüber hinaus bietet diese Möglichkeit der Inanspruchnahme hoher, verteilter Rechenleistung einen ökonomischen Vorteil gegenüber der lokalen Installation vergleichbarer Supercomputer.

Weitere Vorteile können sich auf der Ebene der Softwareentwicklung ergeben, denn zwischen Objekten, Prozessen und der Kommunikation wird oft eine klare Struktur erzwungen oder eine objektorientierte Umgebung gefordert.[303] Dies erleichtert die „Portierung auf verschiedene Umgebungen und verbirgt die dahinterliegende Komplexität.“[304]

Während das Ziel, große Leistungsgewinne für einzelne, spezialisierte Anwendungen zu erzielen, als erreicht bezeichnet werden kann, müssen Metacomputing-Systeme ihre

---

[301] Drum /Effiziente Methoden/ 59

[302] Drum /Effiziente Methoden/ 61

[303] Vgl. Drum /Effiziente Methoden/ 61

[304] Drum /Effiziente Methoden/ 61

Tauglichkeit in kommerziellen Umgebungen oder zur Nutzung als transparenten Zugang zu Hochleistungsrechnern noch zeigen.[305]

Dabei bieten Systeme wie SETI@home und Distributed.net aufgrund ihrer einfachen Infrastruktur und Architektur der darauf rechnenden Applikation einige Vorteile. Die Infrastruktur der Hardware beschränkt sich auf eine leistungsstarke Maschine mit einer guten Netzanbindung für die Verteilung der Rechenaufgaben und das Zusammenführen der Ergebnisse. „Alle weiteren Ressourcen sind verteilt und bedürfen keiner zentralen Administration. Die Applikation muss eine strenge Client/Server Struktur bieten sowie die nötige Logik, um Aufgaben zu unterteilen und Ergebnisse zu sammeln. Diese Einfachheit sorgt für geringere Fehleranfälligkeit des Gesamtsystems und auch für schnelle Applikationsabwicklung.“[306] Die Anwendung solcher Metacomputing-Systeme wird auch zukünftig auf einen kleinen Nischenmarkt beschränkt bleiben.[307] Die Grundvoraussetzung für deren Anwendbarkeit ist neben der für diese Projekte notwendigen und eindeutigen Unterteilung in eine Client-Server Struktur, ein öffentliches Interesse für den jeweiligen Projektinhalt, um die freiwillige Teilnahme sicherzustellen, sowie der Verzicht auf eine vermittelnde Schicht zwischen Hardware und Applikation, so dass jede Applikation das Problem direkt auf die Infrastruktur aufsetzen muss.

WebOS ist eine Integration verschiedener Dienste für weit verteilte Applikationen. Es bietet einen Sicherheitsmechanismus in Form eines redundanten Identifizierungs- und Authentifizierungssystems und ein verteiltes Dateisystem auf der Grundlage des Namensraumes von HTTP.[308]

Das Legion Projekt wird in der Fachliteratur als das am weitesten fortgeschrittene Metacomputing-System bezeichnet, da bereits eine große Anzahl an Anforderungen für weit verteiltes Rechnen erfüllt werden können. Während das Thema der Ressourcen- und Datensicherheit bereits sehr weit entwickelt ist, ergeben sich durch den Verzicht auf eine echte plattformübergreifende Programmiersprache erhebliche Einschränkungen für die Portabilität hinsichtlich der Anwendung.[309]

---

[305] Vgl. Drum /Effiziente Methoden/ 61

[306] Drum /Effiziente Methoden/ 42

[307] Die nachfolgenden Ausführungen sind Vgl. Drum /Effiziente Methoden/ 42, 43 entnommen worden.

[308] Vgl. Drum /Effiziente Methoden/ 32

[309] Vgl. Drum /Effiziente Methoden/ 52

Zusammenfassend ist die Technologie des Metacomputing „nicht nur als ein Nebenprodukt der globalen Vernetzung zu sehen, sondern als eine weiter fortschreitende Spezialisierung der Rechnerlandschaft, mit der für bestimmte Applikationen sehr hohe Rechenleistungen“[310] erzielt werden können. Für einige dieser Applikationen kann Metacomputing einen großen Fortschritt bedeuten. „Für einen Großteil der Applikationen lösen Metacomputer das Problem des Höchstleistungsrechnens nicht, da hier die Anforderungen an die Kommunikationsleistung zu hoch sind.“[311]

11 Ausblick

Die Entwicklung des Metacomputing beschleunigt sich dank des sich kontinuierlich ausbreitenden Internets und der objektorientierten Programmiersprache Java zusehends.[312] Den wohl wichtigsten Faktor für den Erfolg einer Metacomputing-Umgebung stellt der Aspekt der Sicherheit dar. Das Risiko, hoch sensible oder sogar unternehmensinterne Daten zu Auswertungszwecken an unbekannte Rechner in die Hände fremder Personen zu schicken, wird gegenwärtig von potentiellen Teilnehmern als sehr riskant bezeichnet. Aufgrund dieser Einschätzung entscheiden sich Firmen derzeit überwiegend noch gegen verteiltes Rechnen und für eigene Superrechner. In Bereichen des wissenschaftlich-technischen Rechnens werden Metacomputing-Systeme für geeignete Anwendungen jedoch auch in Zukunft eine wichtige Rolle spielen.

„Die Realisierung eines virtuellen Hochleistungsrechners, der sich dem Benutzer oder der Applikation wie ein einzelner Rechner darstellt“[313] und somit dem Ziel des Metacomputing, einer Gleichstellung von Computernetz und Stromnetz, sehr nahe kommt, ist jedoch weniger ein technisches als ein logistisches Problem[314]. Die Entwicklung von „Schnittstellen zu anderen Metacomputer-Architekturen für die Bereitstellung der eigenen Dienste und die Nutzung der Dienste fremder Architekturen für eigene Anwendungen wird ein weiterer Schritt sein, um die existierenden Metacomputer zu einer globalen Recheninfrastruktur zu verbinden“[315]. Eine wesentliche Voraussetzung dafür ist die Entwicklung standardisierter Protokolle, um unabhängig vom Metacomputer auf die

---

[310] Drum /Effiziente Methoden/ 127

[311] Drum /Effiziente Methoden/ 127, 128

[312] Die nach folgenden Ausführungen sind Vgl. Schurz /Gridcomputing/ entnommen worden.

[313] Drum /Effiziente Methoden/ 128

[314] Vgl. Schurz /Gridcomputing/, Vgl. Drum /Effiziente Methoden/ 128

[315] Drum /Effiziente Methoden/ 128

gewünschten Dienste zugreifen zu können. Weitere Gegenstände der Forschung können im Bereich der Skalierbarkeit und der Topologie, die den Systemen zugrunde liegt, gesehen werden, um eine effiziente Kommunikation gewährleisten zu können. Darüber hinaus sollte kontinuierlich an einer Verbesserung der Benutzeroberfläche gearbeitet werden, um die Handhabung der Systeme sowohl für die Administration als auch die Benutzung zu vereinfachen.

Letztendlich hängt die Durchsetzung des Metacomputing auf kommerzieller Ebene entscheidend von der Umsetzung geeigneter Sicherheitsmechanismen ab.

12 Literaturverzeichnis

Albertin /Cracking/

Michael Albertin: Distributed Computing and Cracking.
http://informatik.hsr.ch/Content/Gruppen/Stud/Faecherkatalog/Infosem/Vortraege/SS01/Resultate/Distributed_computing_skript.pdf, Abruf am 2003-10-05.

Anderson et. al. /An Experiment/

David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, Dan Werthimer: SETI@home: An Experiment in Public-Resource Computing.
http://setiathome.berkeley.edu/cacm/cacm.html, Abruf am 2003-07-21.

Andrews et al. /Saguaro/

Gregory R. Andrews, Richard D. Schlichting, Roger Hayes, Titus D.M. Purdin: The design of the Saguaro distributed operating system.
IEEE Transactions on Software Engineering, 13, 1, pp. 104-118.
January 1987

Bechler et al. /Sicherheitskonzept/

Marc Bechler, Achim Hauck, Daniel Müller, Frank Pählke, Lars Wolf: Ein Sicherheitskonzept für clusterbasierte Ad-hoc-Netze.
http://www.ibr.cs.tu-bs.de/users/bechler/myPublications/marc_BHP+02.pdf, Abruf am 2003-08-13.

Belani et. al. /CRISIS Architecture/

Eshwar Belani, Amin Vahdat, Thomas Anderson, Michael Dahlin: The CRISIS Wide Area Security Architecture.
http://www.usenix.org/publications/library/proceedings/sec98/full_papers/belani/belani.pdf, Abruf am 2003-10-19.

Bode /Cluster/

Bode: Cluster. In: VDI-Lexikon Informatik und Kommunikationstechnik.
Berlin, Heidelberg, New York, 2. Auflage, 1999, S. 139-140.

Böszörmenyi /Betriebssysteme/

Laszlo Böszörmenyi: Betriebssysteme - Verteilte Betriebssysteme.
http://bscwpub.itec-e.uni-klu.ac.at/pub/bscw.cgi/d153/C.Verteilte%20Betriebssysteme, Abruf am 2003-09-18.

Casanova, Dongarra /A Network Enabled Server/

Henri Casanova, Jack Dongarra: NetSolve: A Network-Enabled Server for Solving Computational Science Problems. In: The International Journal of SupercomputerApplications and High Performance Computing.
Nr. 11, 1997, S. 212–223.

Coulouris, Dollimore, Kindberg /Concepts and Design/

George Coulouris, Jean Dollimore, Tim Kindberg: Distributed Systems - Concepts and Design.
Nr. 2, Wokingham, England u.a. 1994, S. 1-26.

Drake /SETI@home-Erfolge/

Frank Drake: Wir dürfen noch gar keinen Erfolg haben!
http://www.morgenwelt.de/wissenschaft/991206-drake.htm, Abruf am 2003-05-12.

Drum /Effiziente Methoden/

Philipp Drum: Effiziente Methoden für global verteiltes wissenschaftliches Rechnen. Dissertation vorgelegt an der Fakultät für Informatik an der Technischen Universität München.
München 2001

Eickerman et al. /Metacomputing in gigabit environments/

Thomas Eickermann, Jörg Henrichs, Michael Resch, Robert Stoy, Roland Völpel: Metacomputing in gigabit environments: networks, tools, and applications.
In: Parallel Computing.
Nr. 24, 1998, S. 1847-1872.

Foster, Kesselmann /Computational Grids/

Ian Foster, Carl Kesselmann: Computational Grid.
http://www.globus.org/research/papers/chapter2.pdf, Abruf am 2003-06-07.

Foster, Kesselmann /Metacomputing Infrasturcture/

Ian Foster, Carl Kesselmann: A Metacomputing Infrasturture Toolkit.
ftp://ftp.globus.org/pub/globus/papers/globus.pdf, Abruf am 2003-06-23.

Foster, Kesselmann, Tuecke / Anatomy of the Grid/

Ian Foster, Carl Kesselmann, Steven Tuecke: The Anatomy of the Grid.
http://www.globus.org/research/papers.html, Abruf am 2003-05-29.

Foster et al. /Physiology of the Grid/

Ian Foster, Cral Kesselmann, Jeffrey M. Nick, Steven Tuecke: The Physiology of the Grid - An Open Grid Service Architecture for Distributed Systems Integration.
http://www.globus.org/research/papers.html, Abruf am 2003-05-27.

Frascaria /Peer-to-Peer/

Kareen Frascaria: Peer-to-Peer: Die Erneuerung des verteilten Rechnens.
http://techupdate.zdnet.de/story/0,,t419-s2107183,00.html, Abruf am 2003-06-22.

Freismuth /P2P-Zukunftsmodell/

Dieter Freismuth: Peer-to-Peer – Eintagsfliege oder Zukunftsmodell?
http://www.iicm.edu/research/seminars/ws_01/freismuth-peer2peer.pdf, Abruf am 2003-06-03.

Garg /Elements/

Vijay K. Garg: Elements of Distributed Computing.
Canada 2002, S. 1-8.

Gehring /NRW-Metacomputing/

Jörn Gehring: Was ist Metacomputing?
http://www.uni-paderborn.de/pc2/nrw-mc/mc_def.htm, Abruf am 2003-05-19.

Gleich /Intergrid/

Clemens Gleich: Vom Internet zum Intergrid.
In: c't - Magazin für Computer Technik.
Nr. 22, 2001, S. 208-210.

Gleich /Gleichschaltung/

Clemens Gleich: Gleichschaltung – Privates Parallelrechnen im Internet.
In: c't - Magazin für Computer Technik.
Nr. 22, 2001, S. 202-207.

Grimshaw /Legion/

Andrew Grimshaw: Legion System Architecture.
http://mufasa.informatik.uni-mannheim.de/lsra/persons/markus/sc2000-tutorial/01-grimshaw-europe.pdf, Abruf am 2003-08-23.

Günther /Einführung in Ad-hoc Netzwerke/

Marco Günther: Einführung in Ad-Hoc-Netzwerke.
http://archiv.tu-chemnitz.de/pub/2002/0038/data/vortrag_hrz_pp.pdf, Abruf am 2003-05-23.

Heinzl /Peer-to-Peer-Netzwerk/

Armin Heinzl: Peer-to-Peer-Netzwerk.
In: Peter Mertens, Andrea Back: Lexikon der Wirtschaftsinformatik. 3. Aufl., Berlin, 1997, S. 308-309.

Herrmann /Grid Computing/

Wolfgang Herrmann: Grid Computing - mehr Vision als Realität.
In: Young Professional.
Nr. 3, 2002, S. 42-43.

Hipschman /Radioteleskop/

Ron Hipschman: SETI@home.
http://www.alein.de/seti/radio_search_3_german.htm, Abruf am 2003-07-17.

Hipschman /Wie funktioniert SETI@home?/

Ron Hipschman: Wie funktioniert SETI@home? - Die Aufgliederung der Daten.
http://www.alein.de/seti/about_seti_at_home_2_german.htm, Abruf am 2003-07-05.

Hohenberger /Sicherheit/

Daniel Hohenberger: Sicherheit in Ad-hoc-Netzen.
http://www.net.informatik.tu-muenchen.de/teaching/current/securityUeb/15ausarbeit.pdf, Abruf am 2003-06-25.

Ihlenfeld /Butterfly/

Jens Ihlenfeld: Butterfly.net: Grid soll Millionen Online-Spieler verbinden. http://www.golem.de/showhigh.php?file=/0205/19733.html&wort[]=butterfly, Abruf am 2003-05-14.

Ihlenfeld /IBM - Größtes Grid/

Jens Ihlenfeld: IBM baut größtes GRID-Netzwerk der Welt. http://www.golem.de/0108/15266.html, Abruf am 2003-05-19.

Jansen / Routingalgorithmen/

Ralph Jansen: Routingalgorithmen für infrastrukturlose Paketfunknetze mit kooperativen mobilen Stationen.
Dissertation vorgelegt am Fachbereich Elektrotechnik und Informatik der Universität Siegen.
Siegen 2002

Kaldewey /Simulation/

Tim Kaldewey: Simulation von Mobile Ad Hoc Networks. http://www.tk.informatik.tu-darmstadt.de/Lehre/ss02/semtele/Folien/Tim%20Kaldewey%20-%20Simulation%20von%20Ad%20hoc%20Netzwerken.pdf, Abruf am 2003-06-14.

Karl /Kommunikation/

Holger Karl: Kommunikation in verteilten Systemen. http://www-tkn.ee.tu-berlin.de/curricula/ss00/kvs/folien/PDF/kap0.pdf, Abruf am 2003-06-03.

Klaß /Kommerzielles Grid-Computing/

Christian Klaß: IBM und Globus ermöglichen kommerzielles Grid-Computing. http://www.golem.de/showhigh.php?file=/0202/18381.html&wort[]=IBM&wort[]=und&wort[]=Globus, Abruf am 2003-06-11.

Köhntopp /Betriebssysteme/

Kristian Köhntopp: Betrübssysteme? http://123.koehntopp.de/kris/artikel/betriebssysteme/index, Abruf am 2003-09-24.

Lange /Alternativen/

Christopf Lange: Alternativen zum Client/Server-Modell: Revolution von unten. In: NetworkWorld Germany. Nr. 9, 2001.

Lazowska /Eden/

Edward D. Lazowska, Henry M. Levy, Guy T. Almes, Michael J. Fischer, Robert J. Fowler, Stephen C. Vestal: The architecture of the Eden system. Proceedings of the 8th SOSP, Operating Systems Review, 15, 5, pp. 148 - 159. December 1981

Leopold /Computing/

Claudia Leopold: Parallel and Distributed Computing - A survey of Models, Paradigms and Approaches.
Canada 2001, S. 1-50.

Leopold /Paralleles und Verteiltes Rechnen/

Claudia Leopold: Paralleles und Verteiltes Rechnen. Emailkorrespondenz vom 2003-05-30.

Maehle /Hochleistungs-PC-Cluster/

Erik Maehle: Hochleistungs-PC-Cluster.
http://www.iti.uni-luebeck.de/Misc/Cebit2000/#Motivation, Abruf am 2003-07-02.

Mariske /World Wide Grid/

Hans Arthur Mariske: Wissen: Willkommen im World Wide Grid.
http://www.ftd.de/tm/it/FTDR58JR4VC.html?nv=cpm, Abruf am 2003-05-13.

McDonald, Znati /A Path Availability Model/

A. Bruce McDonald, Taieb Znati: A Path Availability Model for Wireless Ad-Hoc Networks.
http://www.ee.surrey.ac.uk/Personal/G.Aggelou/PAPERS/wcnc.pdf, Abruf am 2003-06-19.

Meister /Parallelisierungspotential/

Gerd Meister: Untersuchung des Parallelisierungspotentials diskreter ereignisgesteuerter Simulationen am Beispiel der Schaltkreissimulation. Dissertation vorgelegt am Fachbereich Informatik der Technischen Universität Darmstadt.
Darmstadt 1999

Minar /Topologies/

Nelson Minar: Distributed Systems Topologies.
http://www.openp2p.com/pub/a/p2p/2002/01/08/p2p_topologies_pt2.html,
Abruf am 2003-05-16.

Murauer /Zugriffschutz in Computersystemen/

Johann Murauer: Informationsflussorientierte Verfahren zum Zugriffschutz in Computersystemen.
Dissertation vorgelegt am Institut für Informationsverarbeitung und Mikroprozessortechnik (FIM) der Johannes Kepler Universität Linz.
Linz 2001

Orzech /Grid/

Dan Orzech: Getting on the Grid.
http://cin.earthweb.com/news/article.php/10493_998721, Abruf am 2003-05-17.

o.V. /Ad-hoc-Netzwerke/

o.V.: Ad-hoc-Netzwerke.
http://www.ira.uka.de/I3V_HTML/FORSCHUNGSVORHABEN/00281235.htm
Abruf am 2003-07-25.

o.V. /Allgemeines/

o.V.: Allgemeines - Die Suche nach außerirdischer Intelligenz.
http://www.engelmarcus.de/main/projekte/setimain.htm#top,
Abruf am 2003-07-06.

o.V. /Architektur/

o.V.: Legion - Architektur.
http://www.informatik.uni-stuttgart.de/ipvr/vs/de/teaching/ss03/seminars/nsvs/papers/Legion.pdf, Abruf am 2003-09-17.

o.V. /Arecibo/

o.V.: Das Arecibo-Radioteleskop.
http://www.signale.de/arecibo/03.html, Abruf am 2003-07-11.

o.V. /Cache Coherence/

o.V.: What is Cache Coherence.
http://whatis.techtarget.com/definition/0,,sid9_gci211729,00.html,
Abruf 2003-05-14.

o.V. /Computersicherheit/

o.V.: Computersicherheit.
http://de.wikipedia.org/wiki/Computersicherheit, Abruf am 2003-08-29.

o.V. /Distributed Computing?/

o.V.: Was ist Distributed Computing?
http://217.160.138.71/portal/index.php?page=8, Abruf am 2003-05-11.

o.V. /Distributed.net - Technik/

o.V.: Distributed.net - Technik.
http://arnold-j.bei.t-online.de/computer.html, Abruf am 2003-10-25.

o.V. /Einleitung zum Metacomputing/

o.V.: Einleitung zum Metacomputing.
http://www-ds.e-technik.uni-dortmund.de/WEB-D/Forschung/parallel.shtml,
Abruf 2003-06-08.

o.V. /Globus/

o.V.: About the Globus Project.
http://www.globus.org/about/default.asp, Abruf am 2003-06-29.

o.V. /Grid Computing/

o.V.: Verteiltes Rechnen - Grid Computing.
http://www.dlr.de/sc/themen/grid, Abruf am 2003-05-16.

o.V. /Introduction to the Grid/

o.V.: Introduction to the Grid.
http://dast.nlanr.net/Guides/GridandGlobus/Grids.html, Abruf am 2003-06-15.

o.V. /Legion/

o.V.: Legion: A Worldwide Virtual Computer.
http://legion.virginia.edu/, Abruf am 2003-06-30.

o.V. /Management/

o.V.: Management.
http://www.4managers.de/01-Themen/..%5C10-Inhalte%5Casp%5C management(by).asp?hm=1&um=M, Abruf am 2003-05-04.

o.V. /News bei SETI@home/

o.V.: News – Alles neu bei SETI@home.
http://www.heise.de/newsticker/data/hps-02.10.02-000/, Abruf am 2003-06-24.

o.V. /OGR-Project/

o.V.: OGR-Project - What are Golomb Rulers anyway?
http://www.hewgill.com/ogr/, Abruf am 2003-09-21.

o.V. /Optimaler Golomb Maßstab/

o.V.: Was ist eigentlich ein Optimaler Golomb-Maßstab?
http://www.distributed.net//ogr/index.php.de, Abruf am 2003-10-02.

o.V. /OGR-24/

o.V.: Optimaler Golomb Maßstab - Projektdetails.
http://217.160.138.71/portal/index.php?page=41&id=ogr, Abruf am 2003-10-06.

o.V. /Past to Present/

o.V.: Metacomputing: Past to Present.
http://archive.ncsa.uiuc.edu/Cyberia/MetaComp/MetaHistory.html, Abruf am 2003-06-03.

o.V. /Peer-to-Peer/

o.V.: peer-to-peer architecture.
http://www.webopedia.com/TERM/p/peer_to_peer_architecture.html, Abruf am 2003-07-03.

o.V. /Peer-to-Peer working group/

o.V.: What is peer-to-peer?
http://www.p2pwg.org/whatis/index.html, Abruf am 2003-04-14.

o.V. /Projektgruppe/

o.V.: Beschreibung der Projektgruppe Mobile Ad Hoc Netzwerke. http://www.uni-paderborn.de/cs/ag-madh/WWW/PGMANET /beschreibung.html, Abruf am 2003-06-16.

o.V. /Provider-Distributed.net/

o.V.: Beschreibung-Distributed.net. http://217.160.138.71/indexjsredir.php?path=provider.php?id=dist, Abruf am 2003-08-15.

o.V. /P2P-Begriffe/

o.V.: P2P-Begriffe. http://kefk.net/P2P/Grundlagen/Begriffe/index.asp, Abruf am 2003-05-13.

o.V. /RC5-72/

o.V.: RC5-72 - Projektdetails. http:/217.160.138.71/portal/index.php?page=41&id=rc72, Abruf am 2003-09-03.

o.V. /Rechenkraft/

o.V.: Rechenkraft - Projektübersicht. http://www.rechenkraft.de, Abruf am 2003-07-05.

o.V. /Rechenzentrum/

o.V.: Das Internet als Rechenzentrum. http://www.ticketboerse.net/distrib.net/distrib.html

o.V. /Rechnernetze/

o.V.: Rechnernetze - die Höchstleistungsrechner der Zukunft. http://wwwhni.uni-paderborn.de/forschung/rechnernetze.php3, Abruf am 2003-05-07.

o.V. /RSA Security/

o.V.: /RSA Security/ http://www.rsasecurity.com, Abruf am 2003-08-15.

o.V. /SETI/

o.V.: SETI.
http://stud4.tuwien.ac.at/~e9425877/data/s240.html, Abruf am 2003-07-06.

o.V. /SETI@home II/

o.V.: SETI@home II mit BOINC.
http://www.boinc.de/setisouth.htm, Abruf am 2003-07-04.

o.V. /Seti-Team/

o.V.: [eom] Seti-Team.
http://www.eom-clan.de/seti/, Abruf am 2003-07-05.

o.V. /SETI@Home I/

o.V.: SETI@Home I (ohne BOINC)
http://www.boinc.de/seti.htm, Abruf am 2003-07-08.

o.V. /Technische Neuigkeiten/

o.V.: http://www.setigermany.de/technews/ge/tech_news_german1999.htm, Abruf am 2003-05-23.

o.V. /Verschlüsselung/

o.V.: Jede Verschlüsselung ist knackbar!
http://www.ciao.com/distributed.net, Abruf am 2003-10-07.

o.V. /Verteilte Betriebssysteme/

o.V.: Verteilte Betriebssysteme: Probleme und Lösungen (Teil 4) - Betriebssysteme müssen künftig Rechner und Netzwerke steuern.
http://www.computerwoche.de/heftarchiv/1992/19921030/a107192.html, Abruf am 2003-09-29.

o.V. /Was ist SETI@home?/

o.V.: Was ist SETI@home ?
http://home.t-online.de/home/heider.home/c59seti.htm, Abruf am 2003-07-01.

o.V. /What have we found?/

o.V.: What have we found?
http://setiathome.ssl.berkeley.edu/found.html, Abruf am 2003-07-07.

o.V. /X.509-Zertifikat/

o.V.: X.509-Zertifikat.
http://www.ietf.org/rfc/rfc2459.txt, Abruf am 2003-08-24.

Rashid, Robertson /Accent/

Richard F. Rashid, George G. Robertson: Accent: A Communication Oriented Operating System Kernel.
Proceedings of the 8th SOSP, Operating Systems Review, 15, 5, pp. 64-75.
December 1981

Sander /Metacomputer-Architektur/

Volker Sander: Eine Metacomputer-Architektur auf der Basis einer kooperativen Ressourcenverwaltung.
http://www.fz-juelich.de/zam/docs/autoren96/sander.html,
Abruf am 2003-06-03.

Schirmbacher /Metacomputing-Editorial/

Peter Schirmbacher: Metacomputing-Editorial.
http://www.hu-berlin.de/rz/rzmit/rzm5/1.pdf, Abruf am 2003-05-03.

Schneider, Quandt /Grundlagen/

Katja Schneider, Silke Quandt: Grundlagen von verteilten Systemen.
http://www.wiwi.hu-berlin.de/~myra/VSvorlesung/Einfuehrung.html, Abruf am 2003-06-02.

Schoder, Fischbach /Ressourcenmanagement/

Detlef Schoder, Kai Fischbach: Peer-to-Peer-Netzwerke für das Ressourcenmanagement. In: Wirtschaftsinformatik 45.
Nr. 3, 2003, S. 313-323.

Schoder, Weinhardt /Technologien/

Detlef Schoder, Christof Weinhardt: Peer-to-Peer – Technologien, Architekturen und Anwendungen. In: Wirtschaftsinformatik 45.
Nr. 3, 2003, S. 257.

Schröder /Grid Computing/

Lars Schröder: Legion - Seminar Grid Computing.
http://www.ivs.tu-berlin.de/Lehre/SS03/GRID/Material/legion.pdf, Abruf am 2003-09-13.

Schumann /Automatic Performance/

Matthias Schumann: Automatic performance prediction to support cross development of parallel programs. In: Proceedings of the SIGMETRICS Symposium on Parallel and Distributed Tools, S. 88–97.
ACM Press, Mai 1996.

Schurz /Gridcomputing/

Frank Schurz: Grid- oder Metacomputing.
http://www2.informatik.uni-jena.de/cmc/cc_seminar_ss02/Grid.pdf,
Abruf am 2003-09-03.

Sheth /Locus/

Anmol Sheth: The Locus Distributed Operating System.
http://www-csl.cs.colorado.edu/csci5573-f01/lectures/TheLOCUSDistributed OperatingSystem_files/frame.htm, Abruf am 2003-10-27.

Shrivastava, Dixon, Parrington /Reliable Distributed Systems/

Santosh K. Shrivastava, Graeme N. Dixon, Graham D. Parrington: Objects and Actions in Reliable Distributed Systems.
http://www.cs.ncl.ac.uk/research/pubs/articles/papers/568.pdf,
Abruf am 2003-10-28.

Smarr, Catlett /Metacomputing/

Larry Smarr, Charles Catlett: Metacomputing.
In: Communication of ACM.
Nr. 6, 1992, S. 45-52.

Stange /Innovative Rechnerarchitekturen/

Horst Stange: Innovative Rechnerarchitekturen und ihr Einsatz.
http://koenigstein.inf.tu-dresden.de/98/gastvortrag.html, Abruf am 2003-06-14.

Strufe /Peer-to-Peer-Distributionssystem/

Thorsten Strufe: Effizientes Peer-to-Peer-Distributionssystem für multimediale Inhalte.
http://diana.prakinf.tu-ilmenau.de/pub/papers/strufe02Effizientes.pdf, Abruf am 2003-05-11.

Tanenbaum /Computernetzwerke/

Andrew S. Tanenbaum: Computernetzwerke.
3. Aufl., München, 2000, S. 17-19.

Tanenbaum, van Steen /Distributed Systems/

Andrew S. Tanenbaum, Maarten van Steen: Diestributed Systems: Principles and Paradigms.
Upper Saddle River, New Jersey, 2002.

Vahdat /Status of WebOS/

Amin Vahdat: Project Status of WebOS. Emailkorrespondenz vom 2003-10-27.

Vahdat et al. /WebOS/

Amin Vahdat, Thomas Anderson, Michael Dahlin, Eshwar Belani, David Culler, Paul Eastham, Chad Yoshikawa: WebOS: Operating System Services for Wide Area Applications.
http://www.cs.utexas.edu/users/less/publications/research/hpdc98.pdf, Abruf am 2003-09-14.

Wörn, Längle /Betriebssysteme/

Heinz Wörn, Thomas Längle: Betriebssysteme und verteilte Systeme.
http://wwwipr.ira.uka.de/~seyfried/Info/Vorlesungsfolien/Info2/VL11.pdf, Abruf am 2003-10-01.

13 Erklärung

Die vorliegende Arbeit habe ich selbständig und ohne Benutzung anderer als der angegebenen Quellen angefertigt. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Quellen entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Unterschrift: