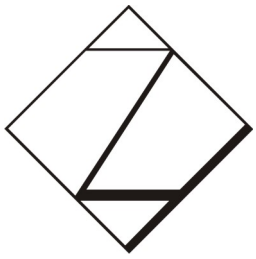


Kommunikationssysteme / Rechnernetze

Aufgabe: Recherche von Tools zur Steuerung eines massiv-parallelen Clusterrechners im Rechenzentrum der Westsächsischen Hochschule Zwickau

Bearbeiter: Danny Christl, Marcel Riedel, Marcus Zelend

Zwickau, im Sommersemester 2007

Inhaltsverzeichnis

1 Was sind Clusterrechner?

1.1 Allgemeines

Ein Computercluster bezeichnet eine Anzahl von vernetzten Computern, die von außen in vielen Fällen als ein einziger Computer angesehen werden können. Die einzelnen Computer in einem Cluster sind in der Regel über ein schnelles Netzwerk verbunden.

Ziel des „Clusterings“ besteht in der Regel in der Erhöhung der der Rechenkapazität (High Performance Computing/HPC-Cluster) oder der Verfügbarkeit (High Availability/HA-Cluster) gegenüber einem einzelnen Computer.

Die einzelnen Computer in einem Cluster nennt man auch „Knoten“ (nodes).

1.2 Homogene vs. Heterogene Cluster

Man unterscheidet homogene und heterogene Cluster. Bei einem homogenen Cluster laufen alle Knoten unter der gleichen Hardware und dem gleichen Betriebssystem. Bei heterogenen Clustern können unterschiedliche Hardware und verschiedene Betriebssysteme eingesetzt werden.

2 High Performance Computing Cluster

2.1 Definition

High Performance Computing Cluster (HPC-Cluster) zielen darauf ab, die Schritte komplexer Aufgaben und Simulationen parallel auf ihren Knoten verarbeiten lassen. Mit diesem Prinzip lässt sich eine enorme Rechenleistung erzielen – diese ergibt sich aus der Summe der Rechenleistung der einzelnen Knoten.

Man kann HPC-Cluster in 2 Kategorien teilen:

- Cluster, bei dem die Knoten einen lokalen Verbund darstellen
- Cluster mit verteilten Knoten – also die Vernetzung arbeitsplatzähnlicher Rechner

Die einzelnen Knoten der Cluster sind meist homogen – also aus gleichartigen Komponenten aufgebaut, was deren Administration vereinfacht.

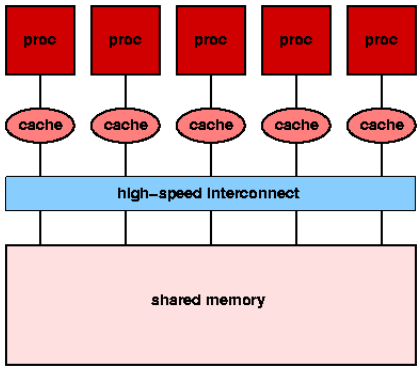
2.2 Funktionsweise

Mithilfe einer spezifischen Software – dem sog. Dekompositionsprogramm – werden die zu verarbeitenden Aufgaben von einem Steuerrechner in einzelne Teilaufgaben zerlegt und durch das Job Management System an die zur Verfügung stehenden Knoten gesendet, die diese parallel bearbeiten. Die Kommunikation zwischen auf verschiedenen Knoten laufenden Job-Teilen geschieht in der Regel mittels Message Passing Interface (MPI), da eine schnelle Kommunikation zwischen einzelnen Prozessen gewünscht ist. Dazu koppelt man die Knoten mit einem schnellen Netzwerk wie z. B. InfiniBand.

2.3 Hardwarekonzept von HPC-Cluster

Im Clusterbereich gibt es 2 grundsätzliche Hardwarekonzepte – Systeme mit Distributed Memory und Systeme mit Shared Memory

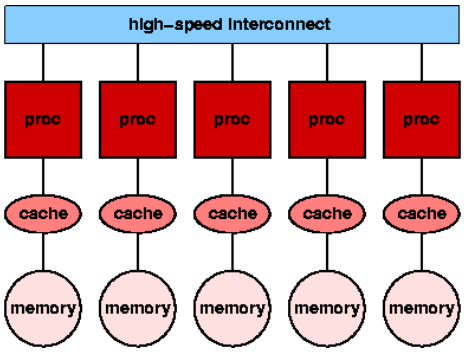
2.3.1 Shared Memory Cluster-Systeme



Bei einem Shared Memory System greifen alle Knoten auf einen gemeinsamen Arbeitsspeicherbereich zu. Ein Verbindungsnetzwerk stellt die Mittel zur Verfügung, die nötig sind um Daten aus dem gemeinsamen Speicher zu den Prozessoren zu übertragen.

Gewöhnlich ist dabei die Zugriffsweise aller Knoten auf den Speicher identisch, sodass die Latenz der Speicherzugriffe für alle Knoten gleich ist. Mit steigender Knotenanzahl erweist sich jedoch selbst das schnellste Verbindungsnetzwerk als Flaschenhals, sodass für Cluster mit hoher Knotenzahl eine Architektur mit Distributed Memory zu favorisieren ist.

2.3.2 Distributed Memory Cluster-Systeme



Bei Architekturen mit Distributed Memory besitzt jeder Knoten einen eigenen Arbeitsspeicher. Somit wird der Flaschenhals der Verbindung zum Speicher aufgelöst. Die einzelnen Knoten greifen nicht auf fremde Speicher zu, sondern tauschen Informationen über Prozesse mittels Nachrichten aus. Über das Netzwerk werden also nun nur noch Steuerdaten sowie Eingabe- und Ergebnisdaten transferiert. Somit eignet sich diese Architektur für Cluster mit einer hohen Knotenzahl, sodass für HPC-Cluster diese Variante zu favorisieren ist. Der Nachteil dieser Architektur ist der erhöhte Aufwand bei der Parallelisierung von Software, da die verteilten Ressourcen explizit vom Programmierer angesprochen und verwaltet werden müssen.

2.3.3 Distributed Shared Memory Cluster-Systeme

Möchte man die Vorteile der weniger aufwändigen SharedMemory-Programmierung auch auf Clustern nutzen, dann kann man auf sogenannte DistributedSharedMemory-Systeme (DSM) zurückgreifen. Die leichtere Programmierung wird aber immer mit einer gewissen Performance-Einbuße, erzeugt durch den DSM-Overhead, erkauft. DSM lässt sich entweder in Hardware oder in Software implementieren.

Für Hardware-Implementierungen gibt es spezielle Hardware, wie z. B. SCI, welche direkt den Speicher adressieren kann und quasi wie ein Memory-Bus wirkt. DSM-Unterstützung ist in der SCI-Hardware fest implementiert und erfordert keine spezielle Unterstützung durch das Betriebssystem. Bei SCI kann für jeden Knoten festgelegt werden, welcher Teil des Speichers nur lokal benutzt werden kann und welcher für das DSM zur Verfügung steht. SCI ist aber relativ neu und noch ziemlich teuer.

Dies ist auch der Grund, weshalb meist die Software-Version bevorzugt wird. Dabei wird der gemeinsame Speicher meistens auf Basis der Speicherseiten des virtuellen Speichers verteilt. Versucht nun ein Knoten auf eine Speicherseite zuzugreifen, welche auf einem anderen Knoten liegt, dann meldet das Betriebssystem einen Seitenfehler, was dem DSM-System mitgeteilt wird. Das DSM-System stellt dann auf dem lokalen Knoten die fehlende Seite entweder durch Kopieren (Replikation) oder Verschiebung (Migration) zur Verfügung.

2.4 Softwarekonzept von HPC-Cluster

Als Betriebssystem kommt fast ausschließlich Linux zum Einsatz, da dieses im High Performance Bereich die breiteste Unterstützung bietet. Spezielle Serverdistributionen sind nicht notwendig. Der Linux-Kernel 2.6 hat sich mittlerweile für den Clustereinsatz bewährt und bietet zudem noch die volle 64Bit-Unterstützung. NFS (Network File System) ist für kleine und mittlere Cluster ausreichend, erst bei wirklich großen Cluster-Systemen sollte man Filesysteme wie z.B. CXFS, Lustre oder Polyserve in Betracht ziehen.

Das Boot from LAN Konzept macht es möglich, dass einzig auf dem Master- bzw. Steuerknoten das Betriebssystem fest installiert ist. Alle anderen Knoten erhalten das Boot-Image vom Masterknoten über das Netzwerk indem sie beim Bootvorgang einen Broadcast über das Netzwerk senden. Voraussetzung dafür ist das Vorhandensein einer PXE-fähigen Netzwerkkarte.

Eine Management-Software, die auf dem Steuerknoten läuft überwacht die Verbindung zu den verbundenen Knoten und ermöglicht deren Administration, welche aber größtenteils automatisch verläuft (z.B. Suche und Einbinden von neuen Knoten).

Die Versorgung der einzelnen Knoten mit Aufgaben erledigt ein Batch-Queue-System. Dieses stellt sicher, dass die einzelnen Knoten gleichmäßig und konstant mit Arbeit versorgt werden.

Abhängig davon ob es sich bei dem Cluster um ein Shared Memory- oder Distributed Memory

System, heterogene oder homogene Hardware handelt ist zu entscheiden, ob PVM (Parallele virtuelle Maschine) oder MPI (Message Passing Interface) unterstützt werden soll. PVM ist ein Software-Paket, welches einen großen Parallelrechner simuliert, und somit auch einen gemeinsamen Speicher. PVM arbeitet auch somit auch mit heterogener Hardware zusammen. Dem gegenüber steht MPI, welches eine Programmierschnittstelle darstellt und den Nachrichtenaustausch bei parallelen Berechnungen auf verteilten Computersystemen beschreibt. MPI arbeitet sowohl unter Architekturen mit Shared Memory als auch Architekturen mit Distributed Memory, allerdings sollte es sich um homogene (gleichartige) Hardware handeln.

2.5 Rechenleistung von HPC-Cluster

Um den Titel des schnellsten Rechnersystems ist in den letzten Jahren in regelrechter Wettkampf ausgebrochen, da immer mehr Einrichtungen auf Cluster setzen um ihre komplexen Probleme zu verarbeiten. Die Organisation TOP500 veröffentlicht halbjährlich eine Zusammenstellung der 500 schnellsten Rechnersysteme. Waren diese bislang meist Supercomputer, so finden sich inzwischen schon 5 Cluster unter den Top15 der Liste:

	Hersteller / Computer	Art	Standort	Leistung (TFlops/s)	Anzahl Knoten
1	IBM eServer Blue Gene Solution / BlueGene L	Super-computer	Lawrence Livermore National Lab, USA	280,6	131072
5	IBM BladeCenter JS21 Cluster, PPC 970 w / Myrinet	Cluster	Barcelona Super-computer Center, Spanien	62,6	10240
6	PowerEdge 1850, 3.6 GHz, Infiniband, Dell	Cluster	NNSA/Sandia National Laboratories, USA	53,0	9024
9	Sun Fire x4600 Cluster, Opteron 2.4/2.6 GHz and ClearSpeed Accelerator, Infiniband, NEC/Sun	Cluster	GSIC Center, Tokyo Institute of Technology, Japan	47,4	11088
11	PowerEdge 1955, 3.0 GHz, Infiniband, Dell	Cluster	Maui High-Performance Computing Center (MHPCC), USA	42,4	5200
12	PowerEdge 1955, 2.66 GHz, Infiniband, Dell	Cluster	Texas Advanced Computing Center, USA	41,5	5200

Tabelle 1: Die leistungsstärksten Cluster im Vergleich zum schnellsten Supercomputer (Stand: 11/2006)

Das schnellste deutsche Clustersystem steht im Forschungszentrum Jülich und besitzt eine Leistung von 37,3TFLOPS, was weltweit Platz 13 bedeutet.

Zum Vergleich: ein derzeit gängiges Desktopsystem hat eine Maximalleistung <10GFLOPS Aufgrund der stetig schneller werdenden Knoten steigt die Gesamtleistung der schnellsten Cluster exponentiell.

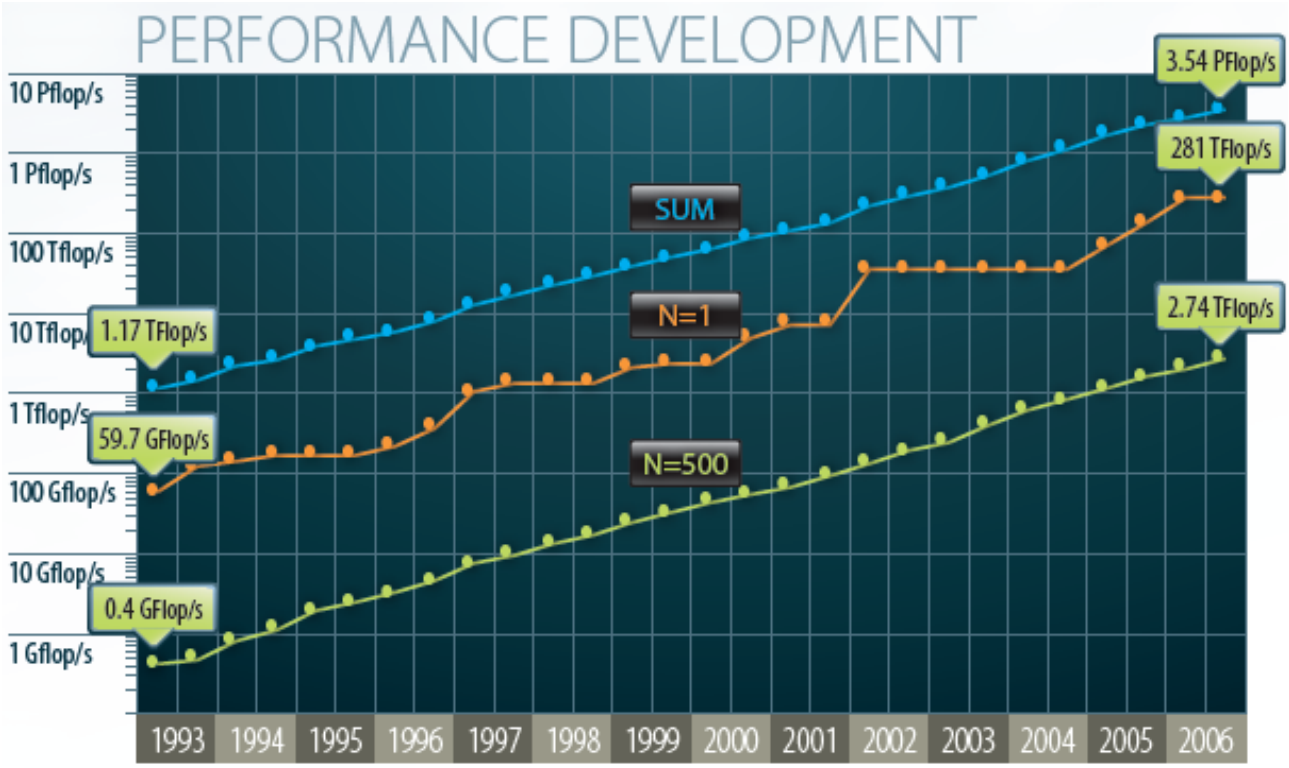


Abbildung 2.1: Entwicklung der Rechnerleistung seit 1993

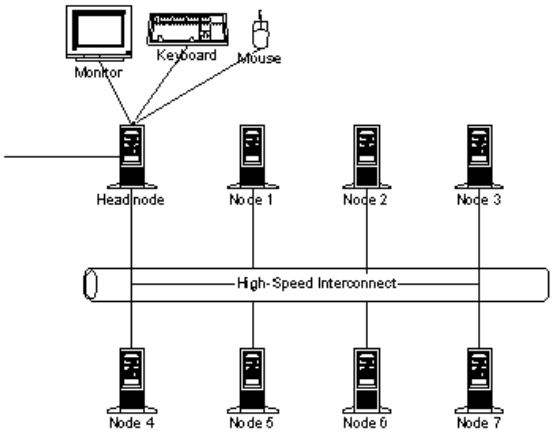
Wie in Abbildung 1 ersichtlich, betrug die Leistung des schnellsten Supercomputer Ende 2006 ca. 281TFLOPS – 281 Billionen Fließkommaoperationen pro Sekunde. In den letzten 13 Jahren entspricht dies einer Steigerung von fast 500000%. Bei einer gleichbleibenden Steigerungsrate wird in ca. einem Jahr die PFLOP-Marke durchbrochen. Eine ähnliche Entwicklung dürfte sich im Clusterbereich abzeichnen.

2.6 Verwendung und Beispiele

Die Verwendung von HPC-Cluster wird vor allem im wissenschaftlichen Rechnen zunehmend notwendig. Sie dienen Hilfsmittel zur Berechnung, Modellierung und Simulation komplexer Systeme und zur Verarbeitung riesiger Messdatenmengen. Derartige Anwendungen finden sich

heute in praktisch allen Bereichen der Natur- und technischen Wissenschaften; typische Anwendungsbereiche sind etwa Meteorologie und Klimatologie, Astro- und Teilchenphysik, Systembiologie, Genetik, Quantenchemie und Strömungsmechanik. Konkrete Anwendungsfälle sind beispielswiese die Berechnung von Klimamodellen oder die Auswertung von Radiosignalen aus dem Weltraum etc. verwendet. Komplexe 3D-Modelle oder aufwändige Animationsfilme werden oftmals auch von HPC-Cluster erstellt, ein solcher Verbund wird dann oftmals auch Renderfarm genannt.

2.7 Beowulf-Cluster als spezielle HPC-Cluster



Beowulf-Cluster ist die Bezeichnung für Cluster, die unter dem freien Betriebssystem Linux oder BSD laufen. Außerdem gibt es auch spezielle Linux-Versionen für derartige Cluster. Beowulf-Cluster zeichnen sich durch ihren Aufbau aus – sie bestehen aus gewöhnlichen, miteinander vernetzten Desktop-PCs. Diese kommunizieren über das TCP/IP-Protokoll miteinander; die Aufteilung der Aufgaben auf die einzelnen Rechner übernimmt auch hier MPI oder PVM (Parallele virtuelle Maschine – ein freies Softwarepaket für verteilte Anwendungen).

Vorteil dieser Cluster-Architektur bietet die problemlose Skalierbarkeit – durch Hinzufügen weiterer Rechner lässt sich Leistungsfähigkeit erhöhen. Somit lassen sich anspruchsvolle Rechenaufgaben mithilfe billiger COTS Hardware (COTS – commercial off-the-shelf – Serienprodukte) lösen; eine Anschaffung von anwendungsbezogener Hardware ist unnötig. Außerdem ist es beim Ausfall einzelner Rechnerknoten möglich, dass diese, während die anderen Knoten weiterlaufen, ausgetauscht werden können.

2.7.1 Der Beowulf-Cluster CLiC

Die TU-Chemnitz betreibt seit dem Jahr 2000 den Beowulf-Cluster „CLiC“, welcher aus 528 Rechnerknoten, zwei Server-Rechner und einem Infrastruktur-Server-Rechner besteht. Die Rechnerknoten haben die folgende Konfiguration:

Abbildung 2.2: Der Beowulf-Cluster „CliC“ im Rechenzentrum der TU Chemnitz

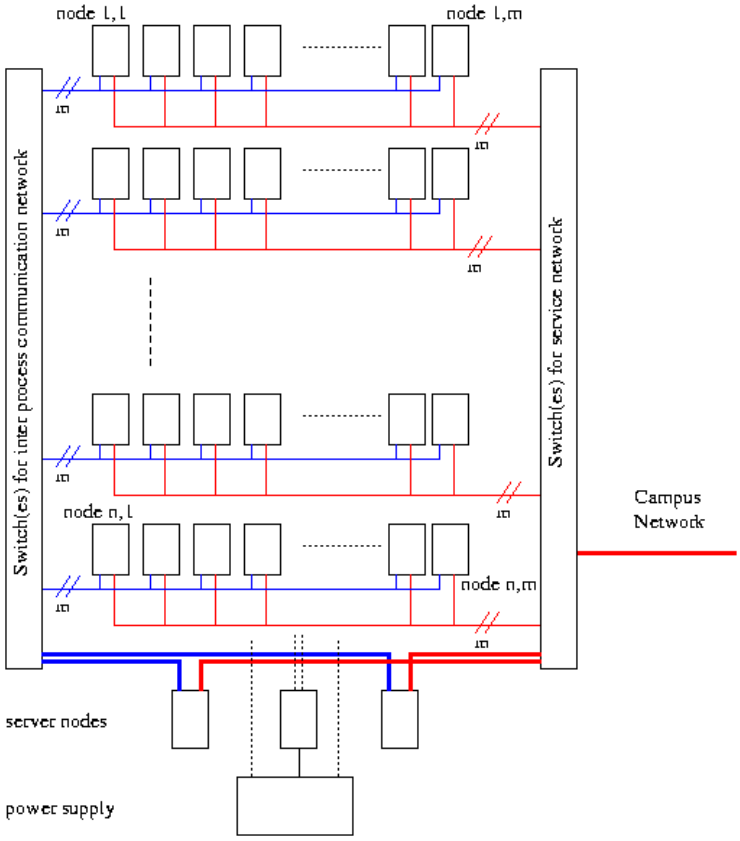


Abbildung 2.3: Netzstruktur des CLiC

Im CLiC sind drei Netzwerke vorhanden, wobei zwei davon die Knoten verbinden und eines ein internes Netzwerk darstellt, welches hauptsächlich für die Steuerung der Infrastruktur (u.a. der Stromzufuhr) der einzelnen Knoten reserviert ist. Die beiden "Knotennetzwerke" sind ein für die Nutzer zugängliches Servicenetzwerk und ein nur innerhalb des Clusters nutzbares Kommunikationsnetzwerk. Somit ist es möglich, dass die Knoten trotz hohem Datenaufkommen für die Systemverwaltung problemlos zumindest auf einem Netzwerk kommunizieren sollten.

Zur Verwaltung der Ressourcen wie z.B. Knotenzahl, Rechenzeitanforderung etc. wird eine angepasste Version des Batch-Systems PBS (Portable Batch System) verwendet. Diese ermögliche dem Nutzer einen interaktiven Zugang zu den Knoten, was beispielsweise für das interaktive Debugging notwendig ist. Auch ist es möglich, durch dynamisches Nachladen von Code in den Ablauf einzugreifen bzw. gewünschte Aktionen zu veranlassen währendem ein Job läuft.

2.8 Technische Daten des geplanten HPC-Clusters im ZKI

Der zur Zeitpunkt der Erstellung dieser Dokumentation im Aufbau befindliche HPC-Cluster im ZKI der Westsächsischen Hochschule Zwickau hat die folgenden technischen Daten:

- 32 Einschübe mit je 2 Tyan Tiger MPX S2466N-4M Boards
- je Board 2 AMD Dual Athlon MP 2600+ Multiprozessor sowie 4x 1 GB DDR-RAM registered ECC von Infineon
- 4 Cisco Catalyst 2950/24XF+ ENET WS-C2950G-24-EI
- 8 Cisco Catalyst 4000, WS-G5484= 1000Base-SX GBIC Interface
- 2 Console Switch Digi DM-32
- 6 MEGWARE ClustSafe Powerswitch
- auf den Knoten ist RedHat Linux Professional 8.0 vorinstalliert

Der Cluster wird von der Firma Megware aufgebaut und eingerichtet.

2.9 Softwarelösungen für HPC-Cluster

2.9.1 OpenMosix

2.9.1.1 Einführung

OpenMosix ist eine freie Implementierung eines High Performance Clusters, der das im Linux-Kernel enthaltene System des Shared Memory zur Parallelisierung nutzt. Mosix wurde ursprünglich für BSD entwickelt und 1997 auf Linux portiert. 2002 wurde Mosix eigenständig und kommerziell vertrieben was dazu führte, dass sich das OpenMosix-Projekt gründete. Dieses Projekt entwickelt seitdem Mosix als freie Software weiter (OpenSource /GPL).

OpenMosix besteht im Wesentlichen aus dem Kernelpatch und den Userland Tools. Diese Werkzeuge dienen dazu den Cluster zu administrieren, Statistiken einzusehen und kümmern sich selbstständig um die Kommunikation mit anderen Knoten.

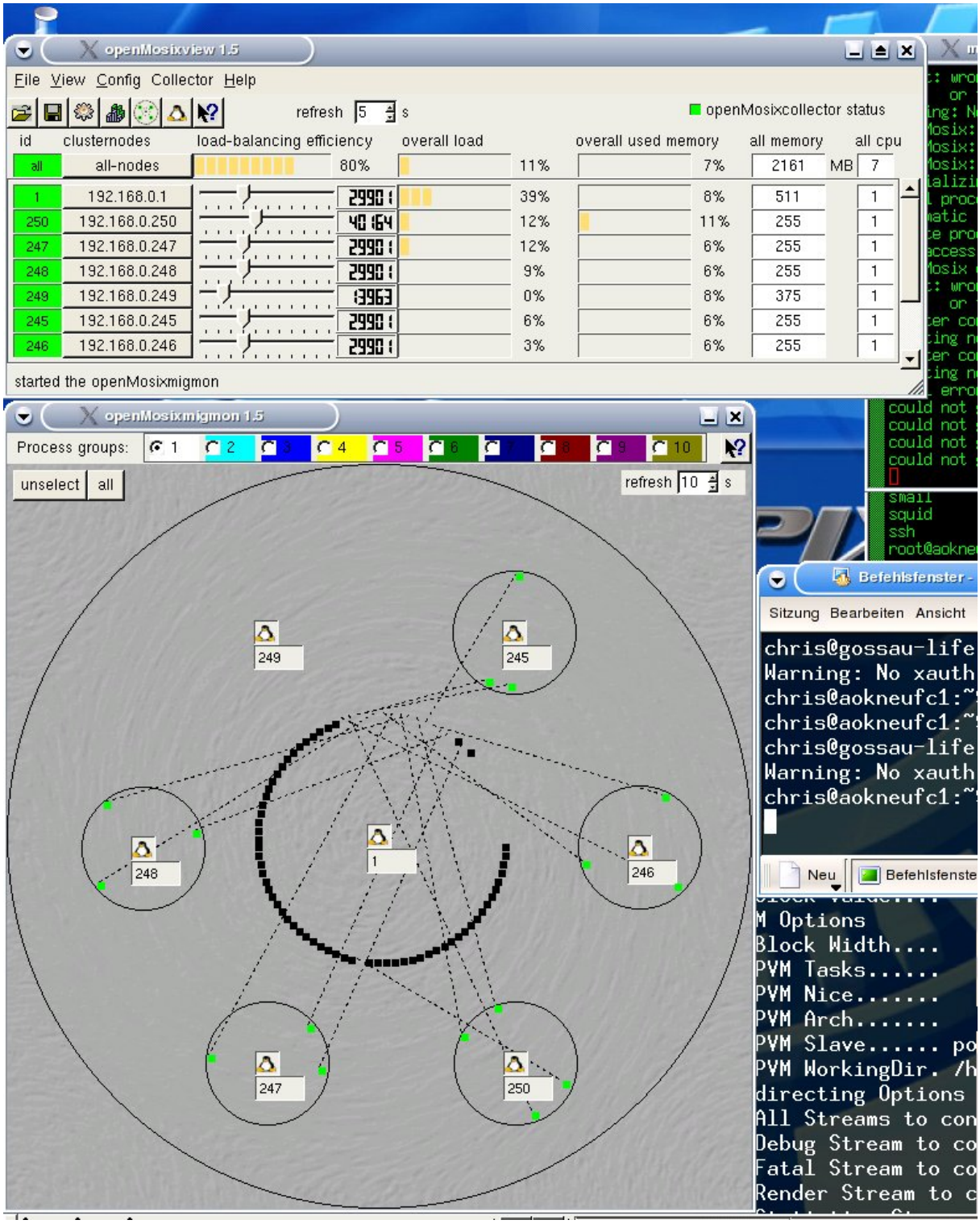


Abbildung 2.4: Clustermanager von OpenMosix unter Knoppix

2.9.1.2 Funktionsweise

OpenMosix arbeitet als SSI-Cluster (Single System Image), d.h. dass alle Knoten logisch zu einer Einheit zusammengeführt werden. Dabei können als Knoten sowohl spezialisierte Clusterrechner sowie normale Personal-Computer verwendet werden. Auf den vereinzelten Knoten muss lediglich ein Linux-Betriebssystem (inkl. des openMosix-Kernelpatches) installiert sein. Ebenso ist es möglich auch beliebige Computer ohne Festplatte zu betreiben, da OpenMosix auch eine PXE-Umgebung zur Verfügung stellt, welche über NFS gebootet wird und sich dann automatisch in den Cluster integriert.

Bei OpenMosix gibt es keinen Master- oder Serverrechner, d.h. jeder eingepflegte Rechner ist gleichberechtigt. Die gleichmäßige Auslastung der Knoten wird dadurch erreicht, dass jeder Node in regelmäßigen Abständen ein Multicast über die noch zur Verfügung stehenden Ressourcen (Prozessorauslastung und freier RAM) sendet. Dadurch weiß jeder Knoten über alle anderen im Netz bescheid und kann damit einen evtl. sehr speicherhungrigen Prozess auf einen anderen Knoten – der momentan nicht ausgelastet ist – auslagern.

Davon bekommen die Nutzer in der Regel nichts mit. Diese arbeiten ganz normal an ihrem Rechner, der für sie wie eine Art Multiprozessor CPU erscheint. Sie erkennen also nicht, dass Prozesse eventuell auf einem anderen Rechner ausgeführt werden.

Die Prozessabarbeitung funktioniert folgendermaßen:

1. Der Benutzer startet einen Prozess ganz normal auf seinem Rechner, wobei eine normale Abarbeitung stattfindet. Dieser Rechner wird im Folgenden als Home-Node bezeichnet.
2. Sofern der Prozess eine Lastgrenze überschreitet prüft der Rechner, ob im Cluster weitere Rechner vorhanden sind die weniger ausgelastet sind.
3. Ist dies der Fall so wird der Prozess auf die gefundenen Rechner verschoben/migriert
4. Alle Speichereiten, die den Prozess im RAM des Home-Nodes darstellen werden kopiert und über das Netz an den anderen Rechner geschickt
5. Der Kernel des Empfangsrechners kopiert nun die Speicherseiten in den RAM
6. Der Prozess wird auf dem anderen Rechner fortgesetzt
7. Auf dem Home-Node bleibt ein so genannter Deputy (Sherrif) zurück, der die Systemaufrufe des Prozesses abfängt.

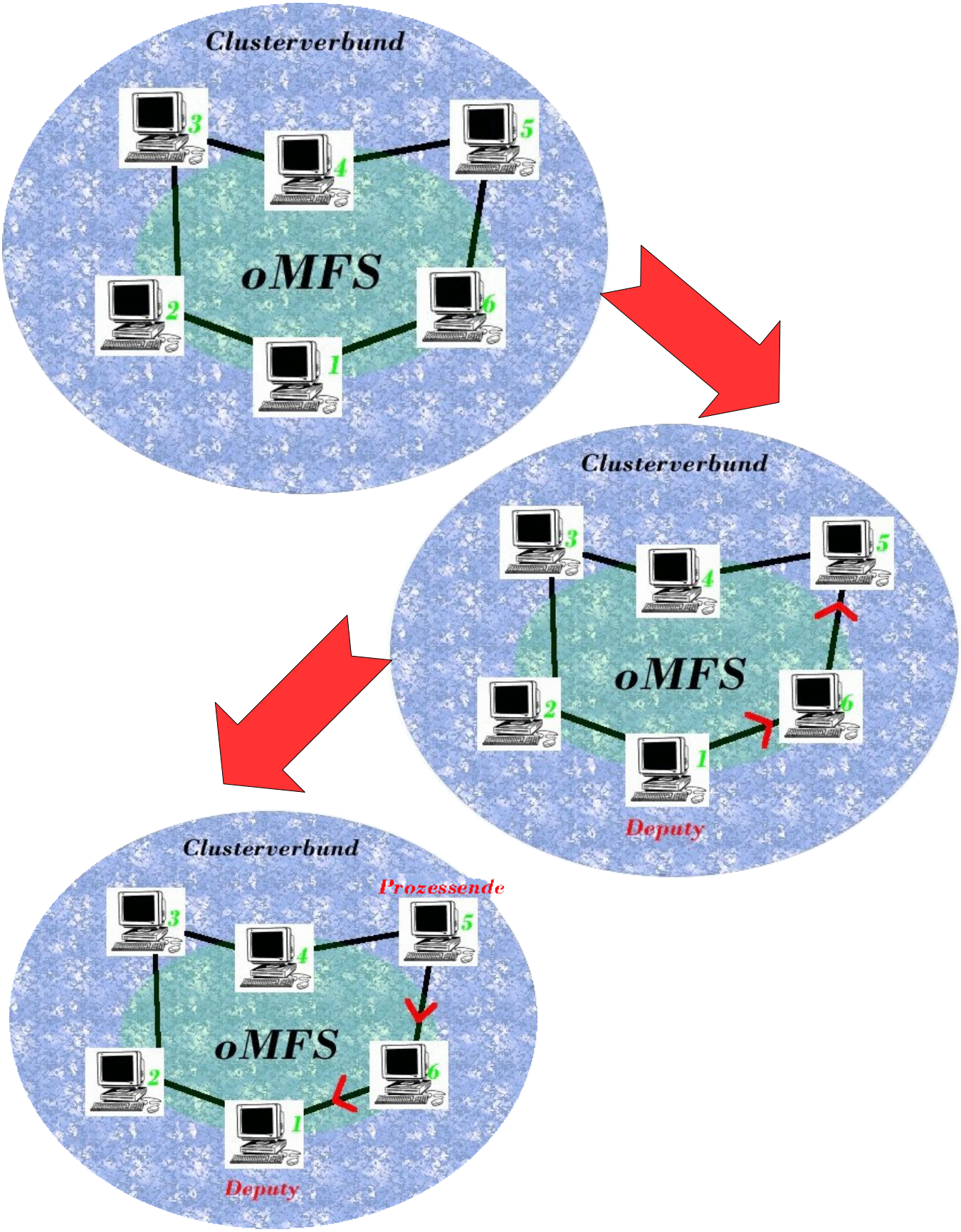


Abbildung 2.5: Schematische Darstellung er Befehlsabarbeitung eines OpenMosix-Clusters

Besonders zu erwähnen ist hierbei das, im Bild erkenntliche, oMFS (OpenMosix-Filesystem). Das hat den Hintergrund, dass bei Prozessen, die einen ständigen Zugriff auf lokale Dateien – sowohl lesend als auch schreibend – benötigen, diese nicht mehr funktionieren wenn sie auf anderen Rechnern migriert sind. Mit Hilfe von oMFS ist es möglich diese Dateien clusterweit bekannt zu machen und zu nutzen, da sonst bei Zugriff jedes Mal der Prozess auf den Home-Node zurück migriert werden müsste, was systemweit zu einem drastischen Ausbremsen führen würde. Dieses Filesystem wurde in der aktuellen Version ersetzt durch das GFS (Global File System).

2.9.1.3 Einsatzgebiete und Fazit

OpenMosix kann sehr vielfältig eingesetzt werden, da nahezu jeder Rechner, unabhängig von der Hardware, integriert werden kann. Jedoch bietet es sich nicht für Systeme an, die eine hohe Verfügbarkeit sichern müssen, da bei Ausfall eines Knotens das ganze System ausfällt. Dies ist bedingt durch die Shared Memory-Architektur, weil das ganze System logisch als ein Einzigstes behandelt wird.

Ein weiterer Vorteil ist, dass jede Linux-Applikation auf dem Cluster lauffähig ist. Dadurch ist der Administrationsaufwand als sehr gering einzuschätzen.

OpenMosix wird hauptsächlich für Aufgaben eingesetzt, die rechenintensiv sind und zeilenweise abgearbeitet/zerlegt werden können. Das wären zum Beispiel wissenschaftliche Berechnungen, wie Wettervorhersagen.

Als Fazit ist zu sagen, dass diese Technologie für das vorliegende System eher ungeeignet ist, da Shared Memory Systeme bei einer großen Prozessoranzahl (hier: 128) unüberschaubar werden und durch die ständige Kommunikation der Knoten das Netzwerk bereits schon sehr ausgelastet ist, was wiederum zu erheblichen Performanceeinbußen führt. Deshalb ist er hier ratsamer einen Clustermanager einzusetzen, welcher die Distributed Memory Architektur unterstützt, da alle benötigten Vorraussetzungen vorhanden und die Leistungseinbußen hierbei gering sind.

2.9.2 OSCAR

2.9.2.1 Definition und Zielsetzung

OSCAR steht für Open Source Cluster Application Resources. Es ist eine Sammlung von Softwarepaketen zum einfachen Aufbau von Clustern auf Basis von PVM/MPI. Sie enthält

umfangreiche Werkzeuge zum Clustermanagement sowie zur Analyse.

OSCAR ist ein Projekt der Open Cluster Group, welche versucht, die Entwicklungen rund um freie Software für den Clustereinsatz zu koordinieren und neue Entwicklungen anzustoßen.

Es hat zum Ziel, die Ressourcen für einen einfach zu installierenden, einfach zu wartenden und einfach zu gebrauchenden Cluster bereit zu stellen. OSCAR implementiert die aktuellen „best-knownpractices“ für High-Performance Cluster.

OSCAR ist für verschiedene Linux-Distributionen benutzbar. Das Paket besteht aus RPMs, Perl-Scripts, Bibliotheken und einigen Werkzeugen, die benötigt werden, um einen Cluster mittlerer Größe zu erstellen.

Die Clustergröße sollte idealerweise zwischen vier und 100 Knoten liegen. Der große Vorteil von OSCAR besteht darin, dass man nicht selbst alle Tools im Internet zusammensuchen muss. OSCAR wird ständig weiterentwickelt, so dass man nicht mit veralteten Tools zurückgelassen wird.

OSCAR ist für all diejenigen das perfekte Tool, die nicht eine spezielle Clusterinstallation oder mehr als 100 Knoten betreiben möchten. Es ermöglicht einen relativ einfachen Einstieg ins Cluster-Computing, auch weil es die Fähigkeit besitzt, in heterogenen Umgebungen zu laufen, da PVM mitgeliefert wird.

2.9.2.2 Komponenten (Auswahl)

2.9.2.2.1 Cluster Command and Control (C3)

C3 ist ein Interface zum Management von Cluster. Es wurde ursprünglich zur Verwaltung des HighTORC-Linux-Clusters (Oak Ridge) entwickelt. Es enthält die folgenden kommandozeilenbasierte Tools, die das Cluster-Management effektiver und schneller machen sollen:

Toolname	Zweck
cpushimage	Verteilen von Systemimages auf den Cluster-Nodes
cpush	Verteilen von Dateien auf den Cluster-Nodes
crm	Clusterweites Löschen von Dateien
cget	Kopieren von Dateien von den Cluster-Nodes auf den Master-Node
cshutdown	Herunterfahren der Cluster-Nodes
ckill	Einen Prozess clusterweit beenden
cexec[s]	Ausführen von Kommandos auf allen Cluster-Nodes
clist	Auflisten sämtlicher Cluster-Nodes
cnum	Liefert Cluster-Name einer bestimmten Knoten-Position

Das aktuelle Release C3 V4.x unterstützt Multithreading und wurde so konzeptioniert, dass es auch größere Cluster effektiv verwalten kann.

2.9.2.2.2 LAM/MPI – Local Area Multi-computer (LAM)

LAM ist eine MPI-Implementierung (MPI – Message Passing Interface) zur Entwicklung von Programmen für Clustersysteme. Es unterstützt MPI V1.2 vollständig und große Teile von MPI V2. Zusätzlich wird noch Debugging ermöglicht. Mit LAM entwickelte Programme sind außerdem voll kompatibel zu anderen MPI-Implementierungen.

Folgende Funktionen werden von LAM unterstützt (Auswahl):

Checkpoint/Restart: Anwendungen können angehalten werden und die aktuellen Daten auf Festplatte ausgelagert werden, sodass sie zu einem späteren Zeitpunkt fortgesetzt werden können.

High Performance Communication: Die Kommunikation zwischen den Knoten kann so eingestellt werden, dass sie eine geringe Latenzzeit bietet und mit einem Minimum an Overhead auskommt – und das bei Geschwindigkeiten im GigabitEthernet-Bereich.

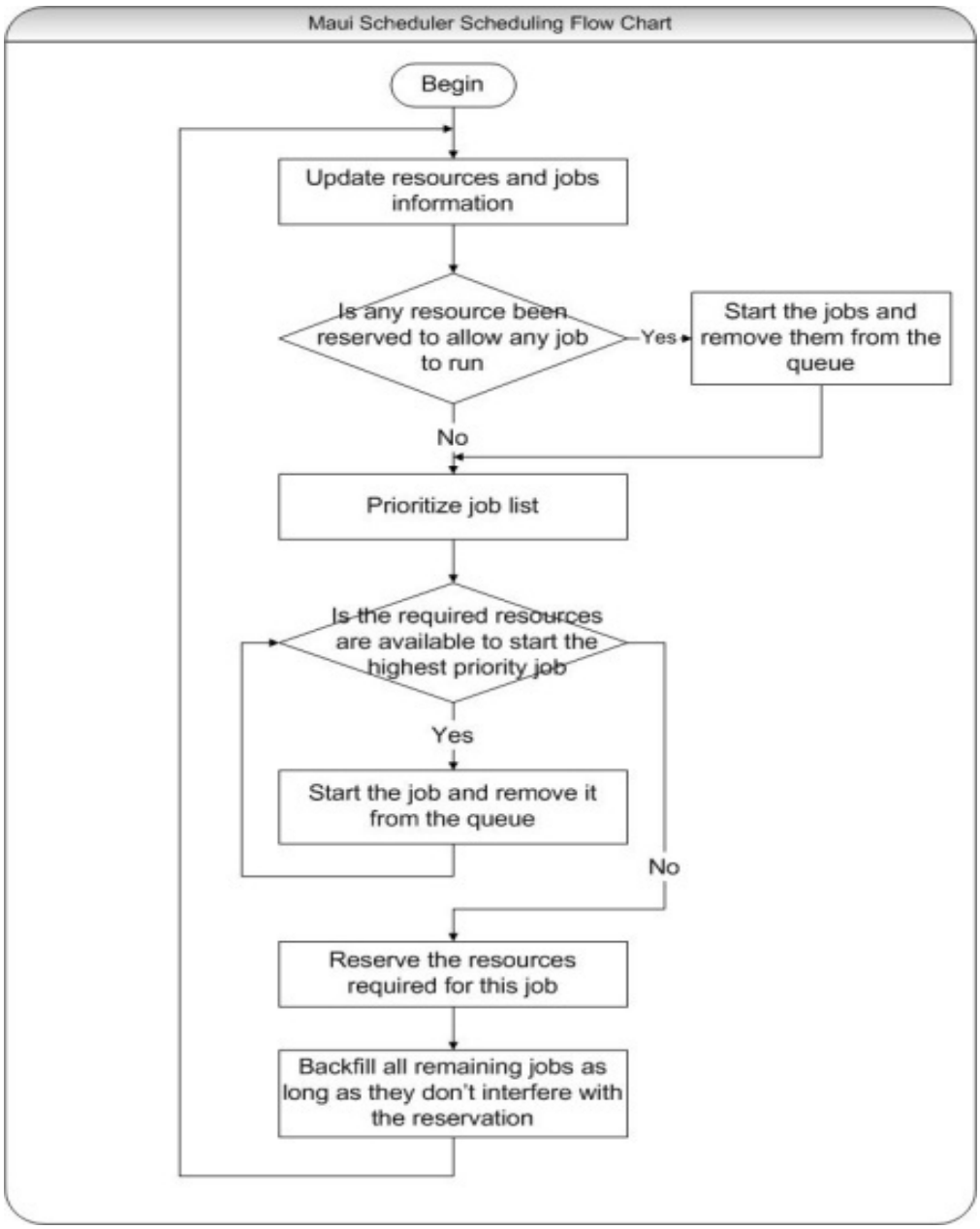
Integration with PBS (Portable Batch System): Mithilfe von OpenPBS bzw. PBS Pro können die Jobs entsprechend einer vorherigen Priorisierung in eine oder mehrere Job-Queues aufgenommen werden. Dafür stehen verschiedene Algorithmen bereit.

Easy Application Debugging: LAM bietet Unterstützung für parallel arbeitende Debugger, selbst für komplizierte MPI-Anwendungen.

PVM – Parallel Virtual Machine (PVM): PVM stellt eine Alternative zu MPI dar. Mithilfe von PVM kann ein Cluster als ein Parallelrechner verwendet werden. Diese Umgebung wird von PVM emuliert und setzt diese auf die Cluster-Architektur auf.

Maui PBS Scheduler: Maui ist ein erweiterter Job-Scheduler für Cluster und Supercomputer. Er ist so optimiert und konfigurierbar, dass sich verschiedene Queue-Richtlinien, dynamische Job-Prioritäten und Gleichberechtigungsalgorithmen einstellen lassen.

Das Job-Management wird wie folgt durchgeführt:



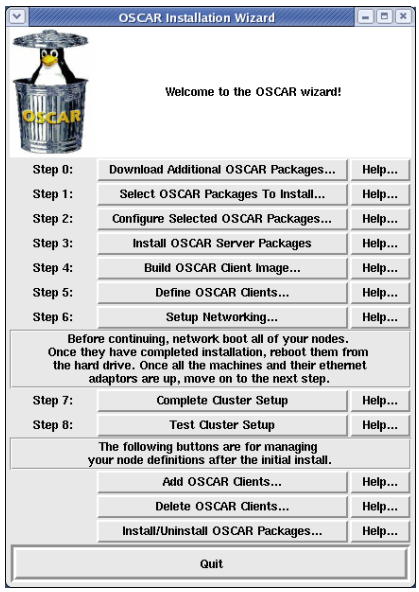
OpenSSH: OpenSSH ist eine freie SSH-Implementierung, welche durch Verschlüsselung eine sichere Kommunikation in offenen Netzwerken, wie z.B. TCP/IP bietet.

OpenSSL: OpenSSL ist ein freiverfügbares Toolkit, mit dem durch Implementierung von SSL- (Secure Sockets Layer) und TLS- (Transport Layer Security) Protokollen sowie einer allgemeinen Kryptografie-Bibliothek ein sicherer Datentransfer möglich wird.

SIS – The System Installation Suite: SIS eine Toolsammlung, die eine automatisierte Installation und Konfiguration von HPC-Clustern ermöglicht. Auch Aktualisierungen sind mittels SIS leicht möglich.

2.9.2.3 Den Cluster mit OSCAR einrichten

Die Installation eines Clusters gestaltet sich mit OSCAR relativ einfach. Man installiert auf dem Master-Knoten eine kompatible Linux-Distribution (z.B. RedHat) und danach die OSCAR-Pakete.



Entweder per Kommandozeile oder mit Hilfe eines intuitiven Konfigurations-Wizard lässt sich dann von da aus der gesamte Cluster konfigurieren. Auch ist es damit möglich mit Hilfe einer Boot-Diskette vom Server aus alle Knoten aufzusetzen. Die Vorbereitung dazu besteht im Wesentlichen aus der Erstellung einer Tabelle aller Knoten mit Namen, MAC-Adresse und Konfigurationsangaben.

2.10 Kerrighed

Kerrighed ist ein Single System Image (SSI)-Betriebssystem für Cluster. Mittels Kerrighed kann ein Cluster aus herkömmlichen PCs als ein symmetrisches Multiprozessorsystem (SMP) angesehen werden.

Ziele von Kerrighed sind einfache Benutzung, eine hohe Performance der Anwendungen, hohe Verfügbarkeit des Clusters, effizientes Ressourcenmanagement und eine individuelle Anpassung des Systems an die verschiedenen Aufgabenbereiche.

Kerrighed ist als eine Erweiterung in ein Linux-Betriebsystem implementiert, genauer als ein Paket von Modulen sowie ein Kernel-Patch.

Kerrighed box kann als wechselweise als Standalone-Version oder als Teil eines Clusters betrieben werden.

Kerrighed ist speziell für wissenschaftliche, numerische Berechnungen gedacht. Das System stützt sich dabei auf OpenMP, MPI oder ein Posix Multithread-Programmmodell.

Weitere Möglichkeiten von Kerrighed sind:

- Clusterweites Prozessmanagment
- Unterstützung für clusterweites Shared Memory
- Cluster File System
- Transparentes Prozess-Checkpointing
- Hohe Verfügbarkeit der User-Anwendungen
- Einstellbare SSI-Features

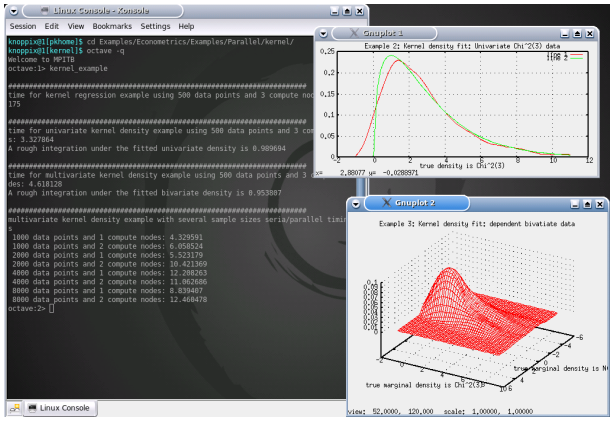
Es gibt zwei Möglichkeiten, einen Kerrighed-Cluster zu starten.

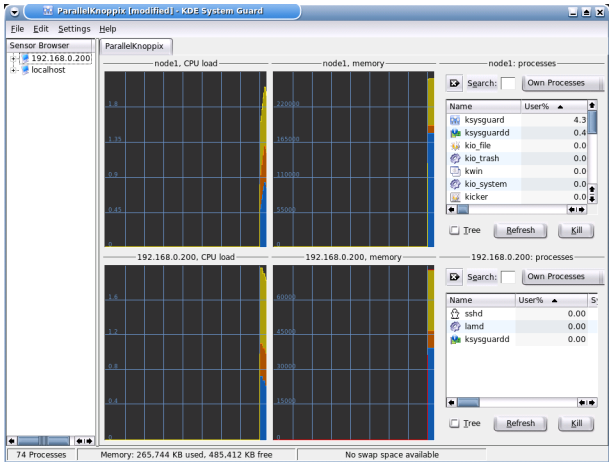
Bei der automatischen Methode wird der Cluster gestartet, sobald eine vorher festgelegte Zahl an Kerrighed-Knoten aktiviert wurden. Der Cluster kann auch manuell gestartet werden.

2.10.1.1 Fazit

Kerrighed ist aufgrund der SMP-Architektur leider nicht für einen Cluster mit sehr vielen CPUs (d.h. mehr als 16) geeignet. Auch ist die Online-Dokumentation der aktuellen Kerrighed-Version 2.x recht dünn. Das System wird aber aktuell weiter entwickelt (Version 2.1.0 vom 05.06.2007).

2.11 ParallelKnoppix





ParallelKnoppix ist ein schneller und einfacher Weg, einen Parallelrechner aufzusetzen. Es ist vor allem für den Einsatz bei Leute geeignet, für die Parallelrechner Neuland sind. Aber auch komplexere Anwendungsgebiete sind kein Problem.

PK wird normalerweise als Live-CD eingesetzt, möglich ist aber auch der Einsatz in einer virtuellen

Umgebung. Der PK-Masterknoten kann dabei in einer virtuellen Maschine laufen, ohne das eigentliche Betriebssystem zu stören. Das virtuelle PK kann dann eingesetzt werden, um über das Netzwerk echte Linux-Maschinen hochzufahren.

Ein PK-Cluster ist „ad hoc“, d.h. auf den Knoten muss keine Software installiert werden. Allerdings befinden sich solche Knoten nach dem Booten stets im Ausgangszustand.

PK stellt einen Cluster von Rechnern zur Verfügung, die für parallele Verarbeitung mittels MPI vorbereitet sind. Dabei sind die Implementierungen openMPI, LAM-MPI und MPICH vorinstalliert und eingerichtet.

Auf der Website zu ParallelKnoppix (http://idea.uab.es/mcreel/ParallelKnoppix/) gibt es ein ausführliches Tutorial zum Aufsetzen eines Clusters in einer virtuellen Maschine, sowie Beispiele für parallele Programmierung.

Aufgrund der sehr guten Unterstützung verschiedener MPI-Implementierung ist PK möglicherweise für einen Einsatz auf einem massiv-parallelen geeignet. In der Online-Dokumentation zu PK sind aber leider keine Angaben hierzu zu finden.

3 High Availability Cluster

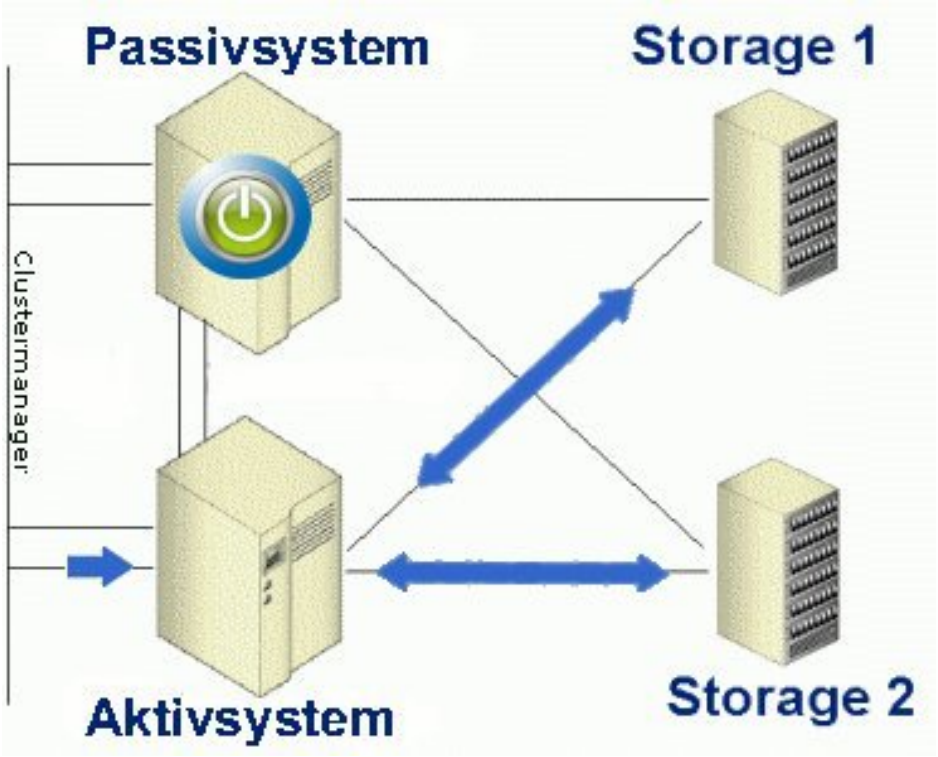
3.1 Definition

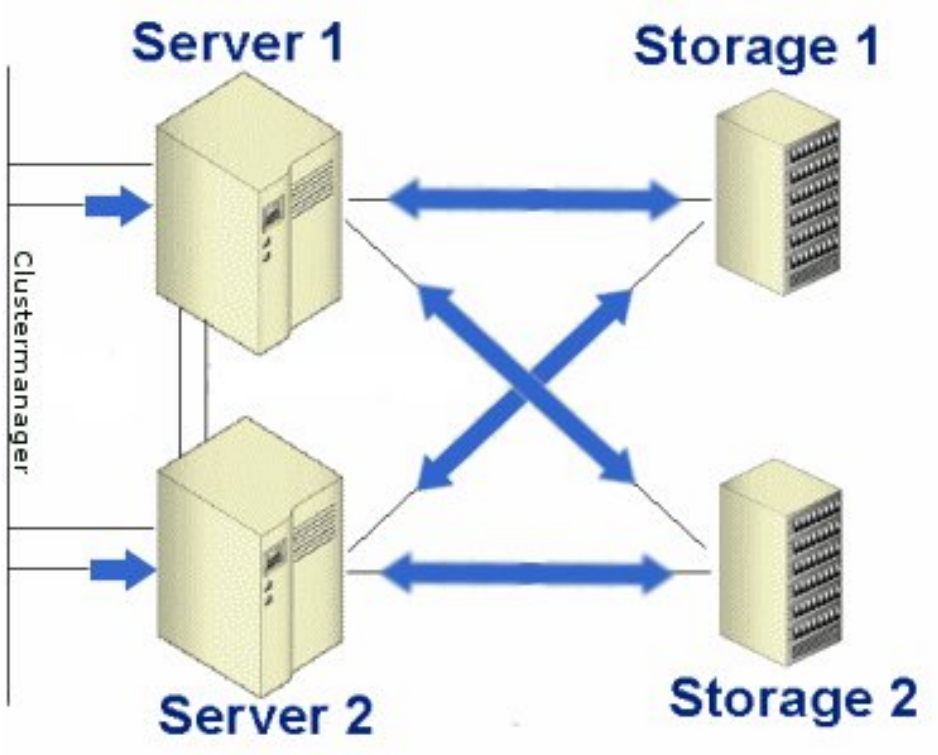
HA-Cluster sind Cluster, die ständig Daten und Dienste zur Verfügung stellen und garantieren eine hohe Ausfallsicherheit (high availibility) im Gesamtsystem. Es gibt 2 Arten von Hochverfügbarkeitsclustern. Zum Einen die Aktiv-/Aktiv-Cluster, wobei der jeweilige Dienst oder die Anwendung auf allen verfügbaren Systemen läuft.Zum Anderen die Aktiv-/Passiv-Cluster oder Fail-Over-Cluster. Hierbei läuft die Anwendung nur auf dem Aktivsystem und das passive System „wartet“ auf den Notfall, also auf den Ausfall des Aktivsystems.

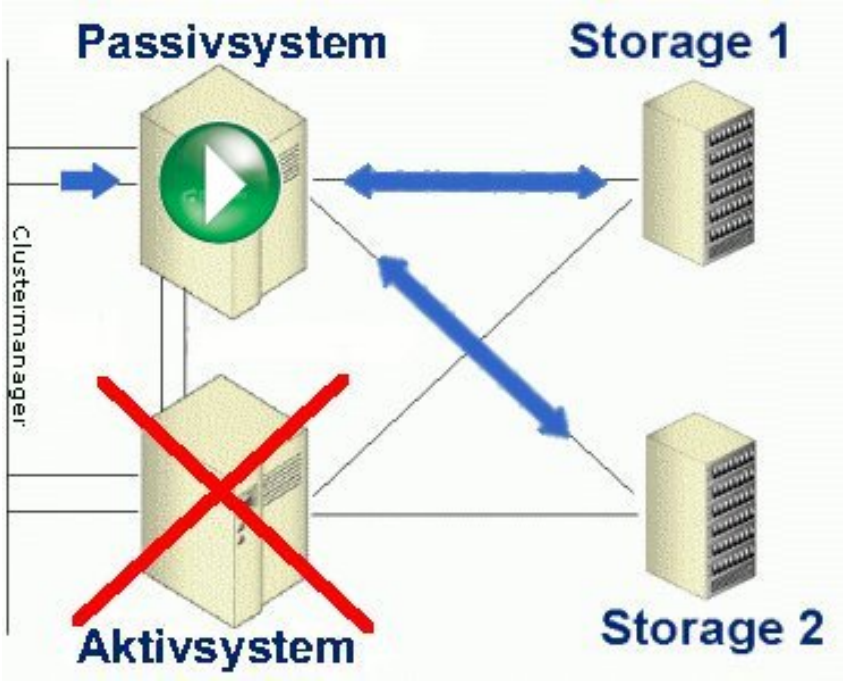
3.2 Funktionsweise

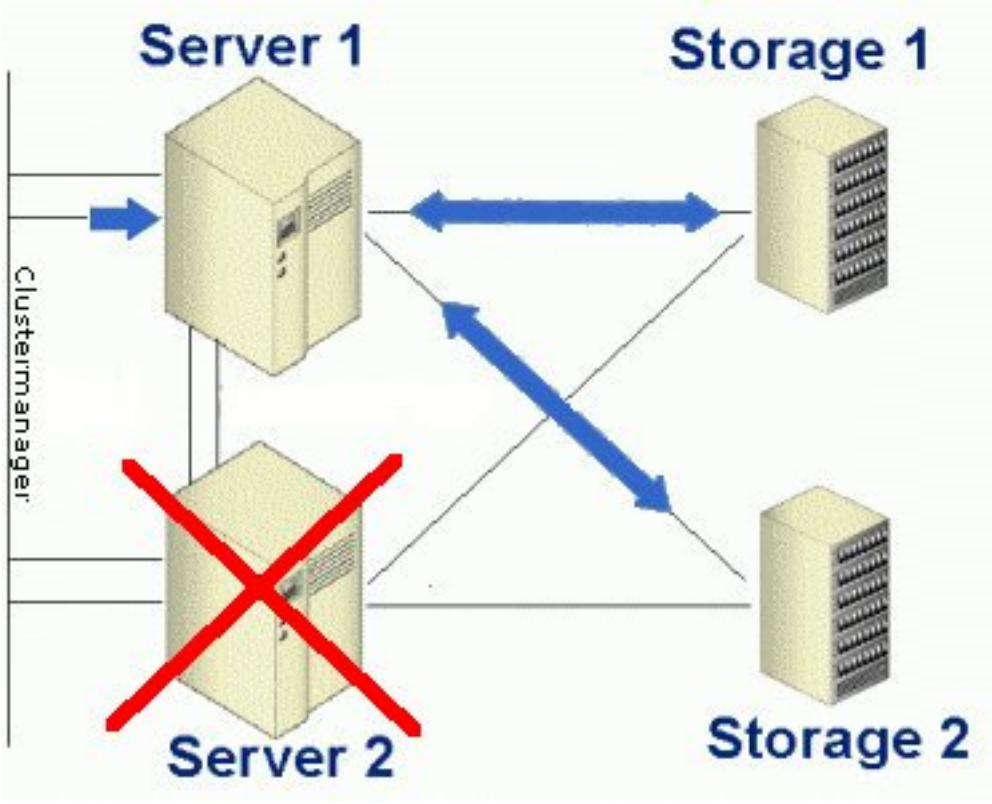
Bei einem HA-Cluster werden – wie bei anderen Clustersystemen auch – mindestens 2 Computer (Knoten) zu einem Verbund zusammengefügt. Das Hauptziel eines HA-Clusters besteht darin Festplatten und Serverausfälle zu minimieren bzw. ganz zu vermeiden. Um dies zu realisieren werden zusätzliche Knoten in das System eingepflegt, die bei Ausfällen eines Rechners die Aufgabe des betroffenen PCs übernehmen und fortsetzen.

Bei einem Aktiv-/Passiv-Cluster laufen die Ersatzknoten ohne Funktion im Hintergrund mit. Bei Ausfall des Aktivsystems bzw. eines seiner Knoten wird das Ersatzsystem aktiviert und übernimmt komplett die Aufgabe des Aktivsystems bis zur Instandsetzung. Hierbei ist ein der Ausfall des Systems nur von sehr kurzer Dauer, da das Passivsystem, welches sich meist im Standby befindet, lediglich gestartet werden muss und der Clustermanager die Aufgaben neu verteilt.


Passivsystem
Storage 1
Clustermanager
Aktivsystem
Storage 2


Server 1
Storage 1
Clustermanager
Server 2
Storage 2

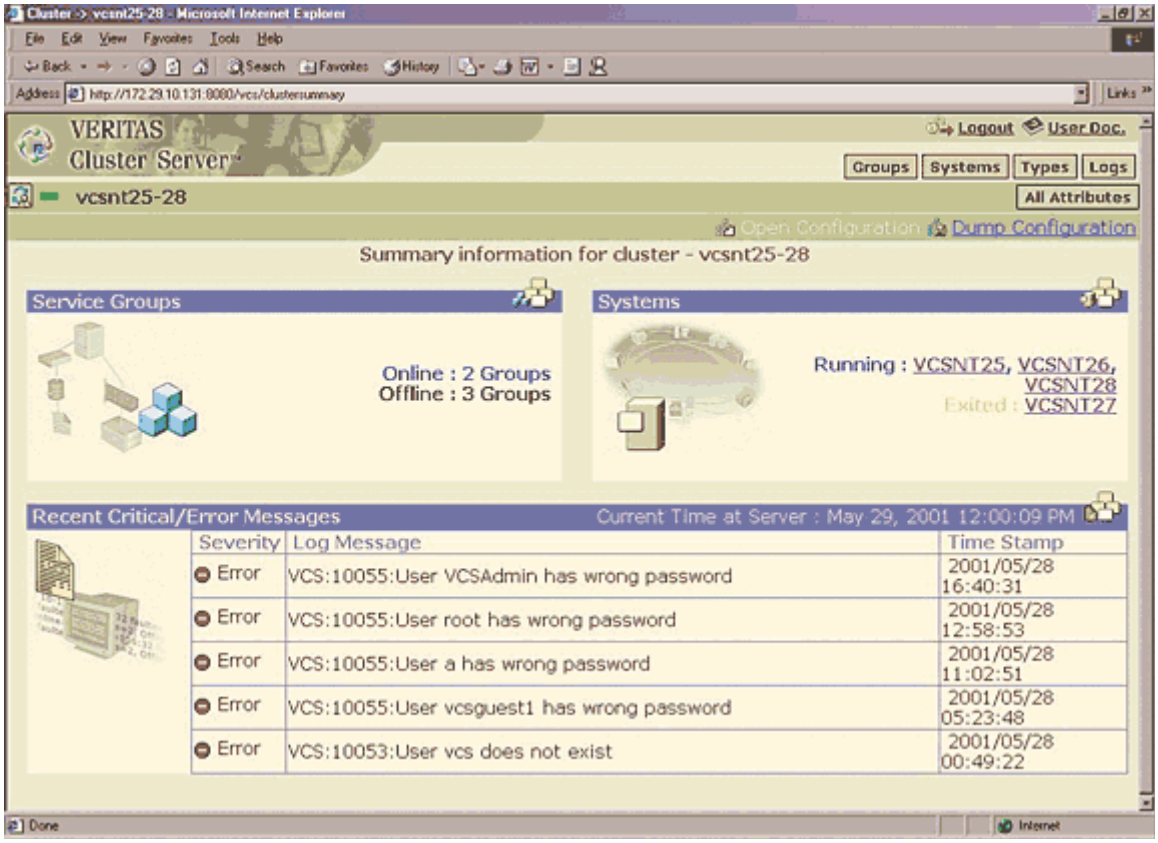




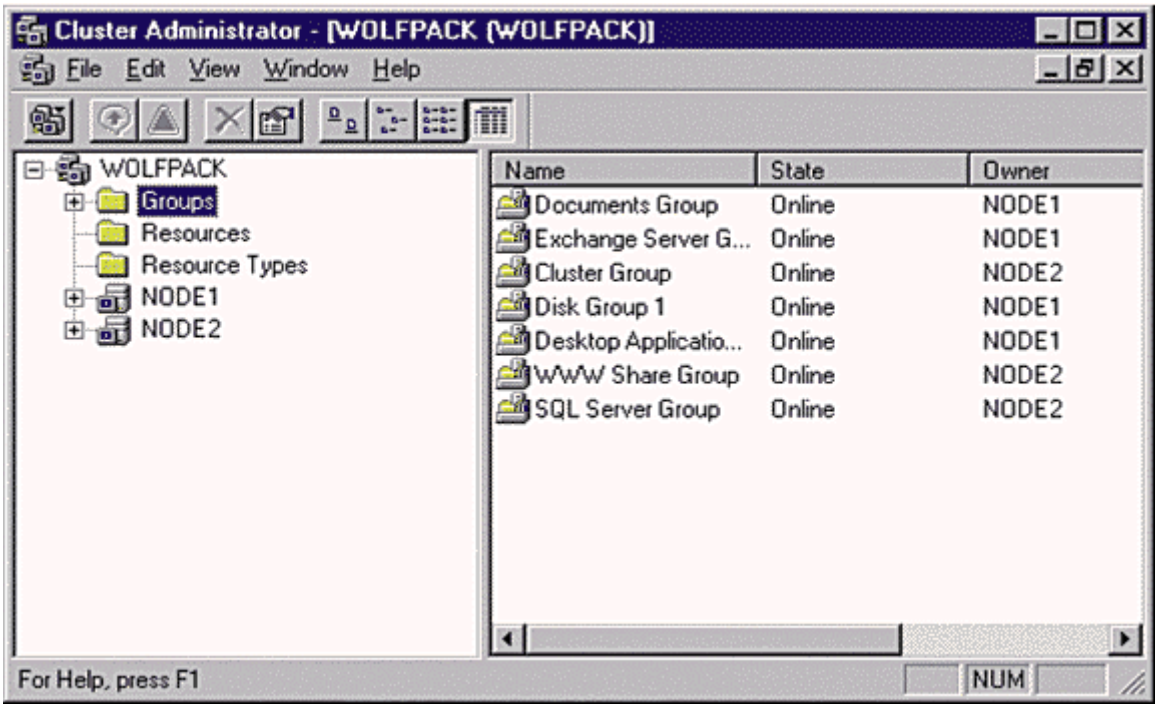
Bei einem Akti-/Aktiv-Cluster laufen immer alle Knoten des Systems gleichzeitig und teilen sich die Aufgaben. Kommt es zum Ausfall eines Rechners im System, wird der Rechner logisch aus dem System entfernt und die restlichen Knoten teilen die Aufgaben des Ausfallrechners, gesteuert durch den Clustermanager.

3.3 Beispiele für Clustermanager

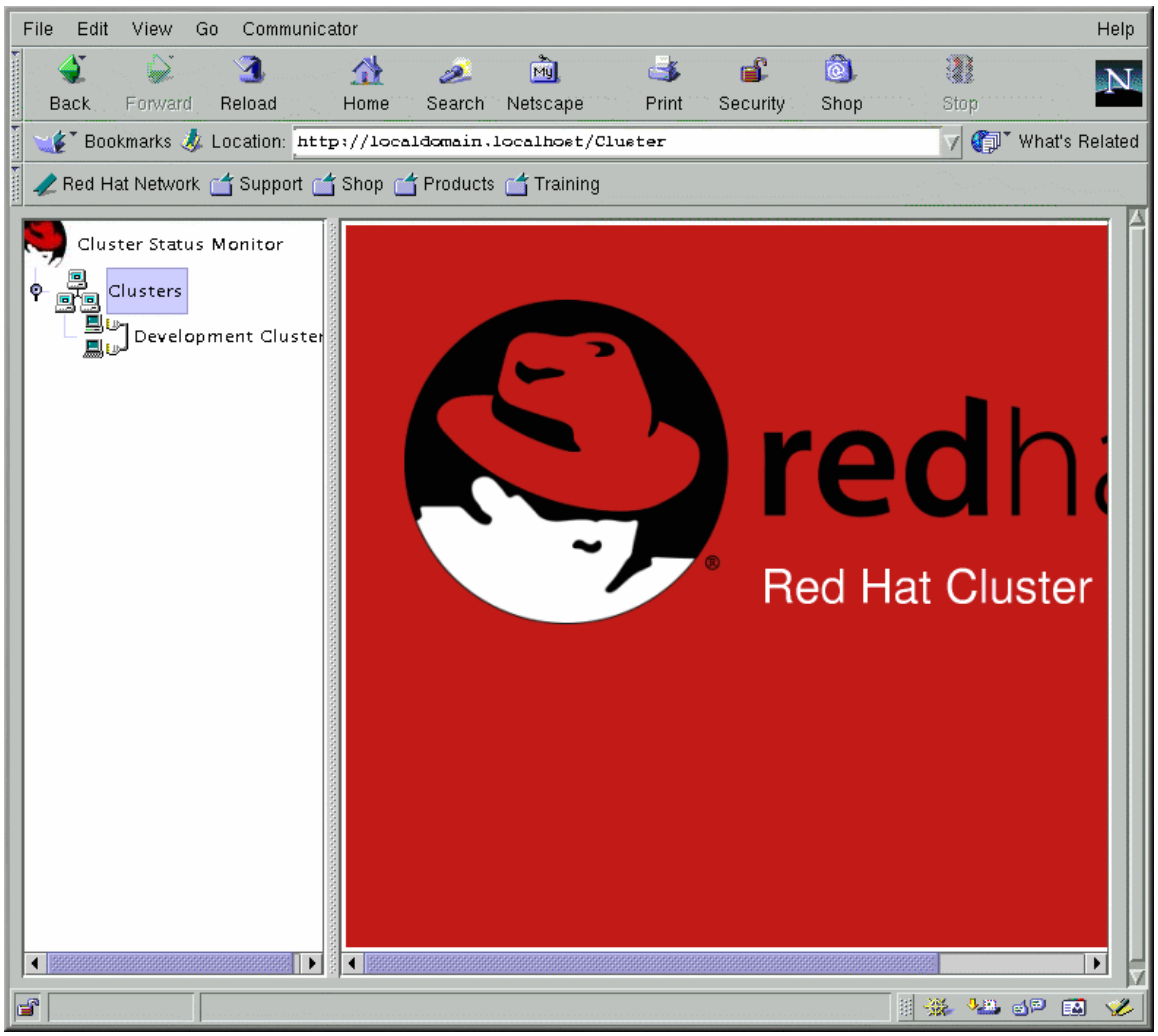
- Veritas Cluster Manager (verschiedene Betriebssysteme, darunter Sun Solaris, HP-UX und Linux)



- Microsoft Cluster Service (in Windows Server 2003 Enterprise Edition bereits enthalten)



- Oracle portable Clusterware (Betriebssystem: alle Plattformen, Unix, Linux, Windows, etc.)
- Red Hat Cluster Manager (Linux)



- SUN Cluster Manager (Betriebssystem: Sun Solaris)
- MC/ServiceGuard (Betriebssystem: HP-UX)
- HACMP (Betriebssystem: AIX)

3.4 Verwendung

HA-Cluster finden größtenteils Verwendung in Servern gekoppelt mit einem NAS (Network Attached Storage), die Dienste (Web-, Datenbank-, FTP- und Mail-Server) im Internet bereit stellen. Dies wäre zum Beispiel ein Webserver eines großen Unternehmens, welches im Internet handelt. Der Ausfall der Internetpräsenz bedeutet nicht nur einen hohen wirtschaftlichen Schaden, in Bezug auf Käufe und Verkäufe bedeutet sondern auch einen immensen Imageschaden.

4 Quellen

http://de.wikipedia.org/wiki/Computercluster

http://de.wikipedia.org/wiki/Failover-Cluster

http://de.wikipedia.org/wiki/Message_Passing_Interface

http://en.wikipedia.org/wiki/Kerrighed

http://en.wikipedia.org/wiki/Message_Passing_Interface

http://en.wikipedia.org/wiki/OpenMosix

http://en.wikipedia.org/wiki/Single_system_image

http://idea.uab.es/mcreel/ParallelKnoppix/

http://linuxwiki.de/OpenMosix

http://openmosix.sourceforge.net/

http://oscar.openclustergroup.org/

http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php

http://www.csm.ornl.gov/torc/C3/

http://www.kerrighed.org/

http://www.lam-mpi.org/

http://www.netlib.org/pvm3/book/node17.html

http://www.openpbs.org/features.html

http://www.storitback.de/index.html?/service/cluster.html

http://www.tu-chemnitz.de/urz/clic/